Furuta Pendulum Virtual Laboratory Experiment

Filippo Badalamenti (Occasional FT, CID: 01998569)

June 6, 2021

This paper will follow the structure presented in the assignment. All the Matlab code and file used will be also provided through the following link: `https://github.com/fil-bad/Furuta_pendulum`.

Part II
Derivation of the model

1 From Lagrange equation to mechanical models

Without going much deeper, we have one main equation for the Lagrangian mechanics:

$$L\left(q\left(t\right),\dot{q}\left(t\right)\right)=T\left(q\left(t\right),\dot{q}\left(t\right)\right)-V\left(q\left(t\right)\right)$$

while the generalized coordinates in our case are: $(q,\dot{q})=\left(\begin{bmatrix}\theta\left(t\right) & \alpha\left(t\right)\end{bmatrix},\begin{bmatrix}\dot{\theta}\left(t\right) & \dot{\alpha}\left(t\right)\end{bmatrix}\right)$. In our case, the Lagrangian equation is:

$$L=\frac{1}{2}J_{arm}\dot{\theta}^{2}+\frac{1}{2}J_{p}\dot{\alpha}^{2}+\frac{1}{2}m_{p}\left(-\cos\left(\theta\right)\sin\left(\alpha\right)\dot{\theta}l_{p}-\sin\left(\theta\right)\cos\left(\alpha\right)\dot{\alpha}l_{p}-\sin\left(\theta\right)\dot{\theta}r\right)^{2}+$$
$$+\frac{1}{2}m_{p}\left(-\sin\left(\theta\right)\sin\left(\alpha\right)\dot{\theta}l_{p}+\cos\left(\theta\right)\cos\left(\alpha\right)\dot{\alpha}l_{p}+\cos\left(\theta\right)\dot{\theta}r\right)^{2}+\frac{1}{2}m_{p}\sin\left(\alpha\right)^{2}\dot{\alpha}^{2}l_{p}^{2}+m_{p}\cos\left(\alpha\right)gl_{p}$$

Finally, the Euler-Lagrange equation can be expressed as:

$$\frac{d}{dt}\frac{\partial L}{\partial\dot{q}}-\frac{\partial L}{\partial q}=\tau$$

For this reason, we get two equations in column:

$$\begin{bmatrix}\frac{d}{dt}\frac{\partial L}{\partial\dot{\theta}}-\frac{\partial L}{\partial\theta}\\ \\ \frac{d}{dt}\frac{\partial L}{\partial\dot{\alpha}}-\frac{\partial L}{\partial\alpha}\end{bmatrix}=\tau$$

Then, we can calculate each partial derivative of the Lagrangian equation, that are:

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{\theta}} = m_p \left(-\cos(\theta)\sin(\alpha)\dot{\theta}l_p - \sin(\theta)\cos(\alpha)\dot{\alpha}l_p - \sin(\theta)\dot{\theta}r\right)\left(\sin(\theta)\sin(\alpha)\dot{\theta}l_p - \cos(\theta)\cos(\alpha)\dot{\alpha}l_p - \cos(\theta)\dot{\theta}r\right) +$$
$$+ m_p \left(-\sin(\theta)\sin(\alpha)\dot{\theta}l_p + \cos(\theta)\cos(\alpha)\dot{\alpha}l_p + \cos(\theta)\dot{\theta}r\right)\left(-\cos(\theta)\sin(\alpha)\dot{\theta}l_p - \sin(\theta)\cos(\alpha)\dot{\alpha}l_p - \sin(\theta)\dot{\theta}r\right) =$$
$$\ldots = \mathbf{0}$$

$$\begin{aligned}
\frac{\partial \boldsymbol{L}}{\partial \dot{\boldsymbol{\theta}}} &= J_{\text{arm}}\dot{\theta} + m_p\left(r\cos(\theta) - l_p\sin(\alpha)\sin(\theta)\right)\left(r\cos(\theta)\dot{\theta} - l_p\sin(\alpha)\sin(\theta)\dot{\theta} + l_p\cos(\alpha)\cos(\theta)\dot{\alpha}\right) + \\
&+ m_p\left(r\sin(\theta) + l_p\sin(\alpha)\cos(\theta)\right)\left(r\sin(\theta)\dot{\theta} + l_p\cos(\alpha)\sin(\theta)\dot{\alpha} + l_p\sin(\alpha)\cos(\theta)\dot{\theta}\right) \\
\frac{d}{dt}\left(\frac{\partial \boldsymbol{L}}{\partial \dot{\boldsymbol{\theta}}}\right) &= J_{arm}\ddot{\theta} - m_p\left(r\cos(\theta) - l_p\sin(\alpha)\sin(\theta)\right)\left(r\sin(\theta)\dot{\theta}^2 - r\cos(\theta)\ddot{\theta} + l_p\sin(\alpha)\cos(\theta)\dot{\alpha}^2 - \right. \\
&- \left. l_p\cos(\alpha)\cos(\theta)\ddot{\alpha} + l_p\sin(\alpha)\cos(\theta)\dot{\theta}^2 + l_p\sin(\alpha)\sin(\theta)\ddot{\theta} + 2l_p\cos(\alpha)\sin(\theta)\dot{\theta}\dot{\alpha}\right) + \\
&+ m_p\left(r\sin(\theta) + l_p\sin(\alpha)\cos(\theta)\right)\left(r\cos(\theta)\dot{\theta}^2 + r\sin(\theta)\ddot{\theta} - l_p\sin(\alpha)\sin(\theta)\dot{\alpha}^2 + \right. \\
&+ \left. l_p\cos(\alpha)\sin(\theta)\ddot{\alpha} - l_p\sin(\alpha)\sin(\theta)\dot{\theta}^2 + l_p\sin(\alpha)\cos(\theta)\ddot{\theta} + 2l_p\cos(\alpha)\cos(\theta)\dot{\theta}\dot{\alpha}\right)
\end{aligned}$$

Once calculated, we can assemble the two equations, and simplifying them we get:

$$\frac{d}{dt}\frac{\partial \boldsymbol{L}}{\partial \dot{\boldsymbol{\theta}}} - \cancel{\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{\theta}}} = J_{arm}\ddot{\theta} - m_p l_p^2\cos(\alpha)^2\ddot{\theta} + m_p l_p^2\ddot{\theta} + m_p l_p r\cos(\alpha)\ddot{\alpha} + m_p r^2\ddot{\theta} - m_p\sin(\alpha)l_p r\dot{\alpha}^2 + 2m_p\sin(\alpha)\dot{\theta}l_p^2\cos(\alpha)\dot{\alpha}$$

The same can be done for the $\alpha(t)$ variable, and reducing the amount of term for each step leads to the equations:

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{\alpha}} = -l_p m_p\left(-\frac{1}{2}l_p\sin(2\alpha)\dot{\theta}^2 + r\sin(\alpha)\dot{\theta}\dot{\alpha} + g\sin(\alpha)\right)$$

$$\frac{\partial \boldsymbol{L}}{\partial \dot{\boldsymbol{\alpha}}} = m_p l_p^2\dot{\alpha} + m_p r\cos(\alpha)l_p\dot{\theta} + J_p\dot{\alpha}$$
$$\frac{d}{dt}\left(\frac{\partial \boldsymbol{L}}{\partial \dot{\boldsymbol{\alpha}}}\right) = m_p l_p^2\ddot{\alpha} + m_p r\cos(\alpha)l_p\ddot{\theta} + J_p\ddot{\alpha} - m_p r\sin(\alpha)\dot{\theta}l_p\dot{\alpha}$$

$$\frac{d}{dt}\frac{\partial \boldsymbol{L}}{\partial \dot{\boldsymbol{\alpha}}} - \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{\alpha}} = \left(m_p l_p^2 + J_p\right)\ddot{\alpha} + m_p r\cos(\alpha)l_p\ddot{\theta} - m_p\cos(\alpha)\sin(\alpha)l_p^2\dot{\theta}^2 + g m_p\sin(\alpha)l_p$$

Equalling the vector:

$$\tau = \begin{bmatrix}\tau_m - B_{arm}\dot{\theta}(t) \\ -B_p\dot{\alpha}(t)\end{bmatrix}, \tau_m = \frac{\eta_g K_g \eta_m K_t\left(V_m - K_g K_m\dot{\theta}\right)}{R_m}$$

to the results found above, finally leads to the nonlinear model expressed in the equations (7) and (8) of the assignment.

2 Matricial model

Our model can be rewritten in the form:

$$D(q(t))\ddot{q}(t) + C(q(t), \dot{q}(t))\dot{q}(t) + g(q(t)) = \tau$$

where the $C\left(q\left(t\right),\dot{q}\left(t\right)\right)$ matrix is not uniquely defined due to mixed products in the equations; a possible representation is given by:

$$D = \begin{bmatrix} m_p r^2 + m_p l_p^2 - m_p l_p^2 \cos\left(\alpha\right)^2 + J_{arm} & m_p \cos\left(\alpha\right) l_p r \\ m_p \cos\left(\alpha\right) l_p r & J_p + m_p l_p^2 \end{bmatrix}$$

$$C = \begin{bmatrix} 2m_p \cos\left(\alpha\right) \dot{\alpha} l_p^2 \sin\left(\alpha\right) & -m_p \sin\left(\alpha\right) \dot{\alpha} l_p r \\ -m_p \cos\left(\alpha\right) \dot{\theta} l_p^2 \sin\left(\alpha\right) & 0 \end{bmatrix} \quad g = \begin{bmatrix} 0 \\ m_p g \sin\left(\alpha\right) l_p \end{bmatrix}$$

Part III
Linear System Analysis

1 Linearizing the model

The matricial equation is not linear; in fact, each matrix depends from the coordinates $q\left(t\right)$ (the angular position of each section of the pendulum) and/or from their derivative $\dot{q}\left(t\right)$ (the angular speed). For this reason, we can linearize the model in its two equilibrium points (as we will see from further analysis), the *downward* position and the *upward* position.

We can then proceed in two ways:

- Neglecting any term with order higher than linear. This can be done using the formula $f\left(x\right) \approx f\left(x_0\right) + f'\left(x_0\right)\left(x - x_0\right)$, that leads for the trigonometric functions to:

  $$\begin{cases} \sin\left(x\right)|_{x_0=0} \approx \sin\left(0\right) + \cos\left(0\right)\left(x-0\right) = x \\ \cos\left(x\right)|_{x_0=0} \approx \cos\left(0\right) + \sin\left(0\right)\left(x-0\right) = 1 \end{cases} \quad \begin{cases} \sin\left(x\right)|_{x_0=\pi} \approx \sin\left(\pi\right) + \cos\left(\pi\right)\left(x-\pi\right) = -x + \pi \\ \cos\left(x\right)|_{x_0=\pi} \approx \cos\left(\pi\right) + \sin\left(\pi\right)\left(x-\pi\right) = -1 \end{cases}$$

  considering the two points later involved.

- Linearize the system around the equilibrium points; in fact, it is legit to assume that both angular speeds are equal to zero – being the position a constant value –, thus allowing us to linearize the rewritten system $\dot{x} = f\left(x,u\right)$ around x_e, following the formula:

  $$\dot{\delta x} = \frac{\partial}{\partial x} f\left(x_e, u_e\right) \delta x + \frac{\partial}{\partial u} f\left(x_e, u_e\right) \delta u$$

  where $\delta x, \delta u$ are the increments, and $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial u}$ are the Jacobian matrices with respect to the state and input vectors, and defining:

  $$A = \left.\frac{\partial}{\partial x} f\left(x,u\right)\right|_{x=x_e} \quad B = \left.\frac{\partial}{\partial u} f\left(x,u\right)\right|_{x=x_e}$$

  we get a linear system in the form $\dot{x} = Ax + Bu$.

Due to a greater familiarity with the second method, the further dissertation will follow the latter one.

1.1 Downward position, $(\theta, \alpha) = (0, 0)$

For this position, we can define the equilibrium point $\begin{pmatrix}\theta & \alpha & \dot{\theta} & \dot{\alpha}\end{pmatrix}^T = \begin{pmatrix}0 & 0 & 0 & 0\end{pmatrix}^T$. Moreover, in order to linearize around this point, we have to rewrite the equation in the form:

$$\dot{x} = \begin{pmatrix}\dot{q} \\ \ddot{q}\end{pmatrix} = \begin{pmatrix} \begin{bmatrix}\dot{\theta} \\ \dot{\alpha}\end{bmatrix} \\ D^{-1}\left(\tau - g - C\begin{bmatrix}\dot{\theta} \\ \dot{\alpha}\end{bmatrix}\right)\end{pmatrix}$$

However, the matrix D must be invertible, and this can be easily verified if $\det(D) \neq 0$:

$$\det(D) = \left(m_p r^2 + m_p l_p^2 - m_p l_p^2 \cos(\alpha)^2 + J_{arm}\right)\left(J_p + m_p l_p^2\right) - \left(m_p \cos(\alpha) l_p r\right)^2$$

but if we consider that all the terms have physical sense when they are strictly positive numbers, we can do some considerations:

- for $\alpha = \pm\frac{\pi}{2}$, the determinant is always sign definite (for everything that makes sense);
- for $\alpha \neq \pm\frac{\pi}{2}$ (and assuming it to be w.l.o.g. $= k\pi$, $k \in \mathbb{Z}$, so that we consider the most negative case), we get:

  $$\left(m_p r^2 + J_{arm}\right)\left(J_p + m_p l_p^2\right) - \left(m_p l_p r\right)^2$$

  so it is enough that the equation, once replaced the real values, it is not fullfilled at the equality with zero.

Once ensured this, we can then write down the state space equations[1]:

$$\begin{bmatrix}x_1(t) & x_2(t) & x_3(t) & x_4(t)\end{bmatrix}^T = \begin{bmatrix}\theta(t) & \alpha(t) & \dot{\theta}(t) & \dot{\alpha}(t)\end{bmatrix}^T$$

so that we can rewrite the equation as:

$$\dot{x} = \begin{pmatrix}\dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4\end{pmatrix} = \begin{pmatrix} \begin{matrix}x_3 \\ x_4\end{matrix} \\ D^{-1}\left(\tau - g - C\begin{bmatrix}x_3 \\ x_4\end{bmatrix}\right)\end{pmatrix}$$

where:

$$D^{-1} = \frac{1}{\det(D)}\begin{bmatrix}D_{22} & -D_{12} \\ -D_{21} & D_{11}\end{bmatrix}$$

and D_{ij} denote the component of the matrix D in the i-row, j-column position.

We can then linearize the system through the Jacobian operator, and then evaluating the resulting matrices in the equilibrium point x_{down} (thanks to the *subs* function in Matlab).

Without writing the full matrix equations (they are difficult to fit in a page), the resulting matrices of the linearized system are:

[1] For simplicity, I will treat (Q1) and (Q2) of Part III at the same moment, writing directly the state space representation.

$$A = \left.\frac{\partial}{\partial x} f(x,u)\right|_{x=x_{down}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{g l_p^2 m_p^2 r}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} & -\frac{\left(m_p l_p^2 + J_p\right)\left(B_{arm} + \frac{\eta_g \eta_m K_g^2 K_m K_t}{R_m}\right)}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} & \frac{B_p l_p m_p r}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} \\ 0 & -\frac{g l_p m_p \left(m_p r^2 + J_{arm}\right)}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} & \frac{l_p m_p r \left(B_{arm} + \frac{\eta_g \eta_m K_g^2 K_m K_t}{R_m}\right)}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} & -\frac{B_p \left(m_p r^2 + J_{arm}\right)}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} \end{bmatrix}$$

$$B = \left.\frac{\partial}{\partial u} f(x,u)\right|_{x=x_{down}} = \begin{bmatrix} 0 \\ 0 \\ \frac{\eta_g \eta_m K_g K_t \left(m_p l_p^2 + J_p\right)}{R_m \left(J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p\right)} \\ -\frac{\eta_g \eta_m K_g K_t l_p m_p r}{R_m \left(J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p\right)} \end{bmatrix}$$

1.2 Upward position, $(\theta, \alpha) = (0, \pi)$

Following the same approach, we have as equilibrium point $\begin{pmatrix} \theta & \alpha & \dot{\theta} & \dot{\alpha} \end{pmatrix}^T = \begin{pmatrix} 0 & \pi & 0 & 0 \end{pmatrix}^T$. Since the state vector that we have to find out is $\begin{bmatrix} x_1(t) & x_2(t) & x_3(t) & x_4(t) \end{bmatrix}^T = \begin{bmatrix} \theta(t) & \alpha(t) - \pi & \dot{\theta}(t) & \dot{\alpha}(t) \end{bmatrix}^T$. This means that we can change the coordinates for α so that, when we will linearize, we do it around the new origin. Another approach is instead to leave the equations unchanged and then linearize around the equilibrium point expressed as function of θ and α, and later consider the equilibrium point in the new coordinates (hence $x_2 = \alpha - \pi \overset{\alpha=\pi}{\Longrightarrow} x_2 = 0, x_2 + \pi = \alpha$).

If we proceed as seen above, the system becomes:

$$A = \left.\frac{\partial}{\partial x} f(x,u)\right|_{x=x_{up}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{g l_p^2 m_p^2 r}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} & -\frac{\left(m_p l_p^2 + J_p\right)\left(B_{arm} + \frac{\eta_g \eta_m K_g^2 K_m K_t}{R_m}\right)}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} & -\frac{B_p l_p m_p r}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} \\ 0 & \frac{g l_p m_p \left(m_p r^2 + J_{arm}\right)}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} & -\frac{l_p m_p r \left(B_{arm} + \frac{\eta_g \eta_m K_g^2 K_m K_t}{R_m}\right)}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} & -\frac{B_p \left(m_p r^2 + J_{arm}\right)}{J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p} \end{bmatrix}$$

$$B = \left.\frac{\partial}{\partial u} f(x,u)\right|_{x=x_{up}} = \begin{bmatrix} 0 \\ 0 \\ \frac{\eta_g \eta_m K_g K_t \left(m_p l_p^2 + J_p\right)}{R_m \left(J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p\right)} \\ \frac{\eta_g \eta_m K_g K_t l_p m_p r}{R_m \left(J_{arm} m_p l_p^2 + J_p m_p r^2 + J_{arm} J_p\right)} \end{bmatrix}$$

Part IV
System Analysis

1 Evaluating A, B matrices and assessing stability

1.1 Downward position

Using the values given in the Table 1 of the assessment, and substituting it in the two matrices, we get:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{5169259020477411}{97834235525000} & -\frac{15506119043916576}{794903163640625} & \frac{2559865848}{3913369421} \\ 0 & -\frac{38382920592829641}{391336942100000} & \frac{15475651065023064}{794903163640625} & -\frac{4751896122}{3913369421} \end{bmatrix} \approx \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 52.837 & -19.507 & 0.654 \\ 0 & -98.082 & 19.469 & -1.214 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ \frac{44598579129216}{1271845061825} \\ -\frac{44510947365024}{1271845061825} \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \\ 35.066 \\ -34.997 \end{bmatrix}$$

Now, to assess the stability property, we have to remember for a linearized system, we have to look at the eigenvalues of the matrix $A = \frac{\partial}{\partial x} f(x_e, u_e)$, for which three possible cases arise:

1. The matrix $\frac{\partial}{\partial x} f(x_e, u_e)$ is Hurwitz. This means that:

$$\forall \lambda_i \in eigs\left(\frac{\partial}{\partial x} f(x_e, u_e)\right) = \{\lambda_1, \ldots, \lambda_n\} : Re\{\lambda_i\} < 0$$

   In this case, the equilibrium is locally asymptotically stable.

2. The matrix $\frac{\partial}{\partial x} f(x_e, u_e)$ is such that:

$$\exists \lambda_i \in eigs\left(\frac{\partial}{\partial x} f(x_e, u_e)\right) = \{\lambda_1, \ldots, \lambda_n\} : Re\{\lambda_i\} > 0$$

   In this case, the equilibrium is unstable.

3. The matrix $\frac{\partial}{\partial x} f(x_e, u_e)$ is such that:

$$\begin{cases} \forall \lambda_i \in eigs\left(\frac{\partial}{\partial x} f(x_e, u_e)\right) = \{\lambda_1, \ldots, \lambda_n\} : Re\{\lambda_i\} \leq 0 \\ \exists \lambda_j : Re\{\lambda_j\} = 0 \end{cases}$$

   In this case, we cannot assert anything; it is logical since we don't have a single point with minimum energy; instead, we have a minimum for every value of θ. The only thing we could eventually assess is that it would be Lyapunov stable, but not asymptotically convergent (in fact, the point of convergence depends from the specifical initial condition).

For the downward position, the eigenvalues are:

$$eigs(A_{down}) = \begin{Bmatrix} 0 \\ -\frac{1222738973915481}{70368744177664} \\ -\frac{3766174288345861}{2251799813685248} + j\frac{976194666914269}{140737488355328} \\ -\frac{3766174288345861}{2251799813685248} - j\frac{976194666914269}{140737488355328} \end{Bmatrix} \approx \begin{Bmatrix} 0 \\ -17.376 \\ -1.673 + j6.936 \\ -1.673 - j6.936 \end{Bmatrix}$$

From the rules analyzed above, we cannot assess any form of stability from linearization.

1.2 Upward position

From the point of view of simply substituting values, we can use the results found for the downward position and change the signs appropriately. Hence, we have:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{5169259020477411}{97834235525000} & -\frac{15506119043916576}{794903163640625} & -\frac{2559865848}{3913369421} \\ 0 & \frac{38382920592829641}{391336942100000} & -\frac{15475651065023064}{794903163640625} & -\frac{4751896122}{3913369421} \end{bmatrix} \approx \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 52.837 & -19.507 & -0.654 \\ 0 & 98.082 & -19.469 & -1.214 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ \frac{44598579129216}{1271845061825} \\ \frac{44510947365024}{1271845061825} \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \\ 35.066 \\ 34.997 \end{bmatrix}$$

Now, assessing stability, and applying the same rules as the previous point, we get:

$$eigs(A_{up}) = \begin{Bmatrix} 0 \\ -\frac{1607239695991555}{70368744177664} \\ \frac{8300623629294261}{1125899906842624} \\ -\frac{5914786364422933}{1125899906842624} \end{Bmatrix} \approx \begin{Bmatrix} 0 \\ -22.840 \\ 7.372 \\ -5.253 \end{Bmatrix}$$

As we expected, the upward position is an unstable equilibrium.

2 Checking controllability

We can verify the controllability of a linear system checking if the $R = \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix} \in \mathbb{R}^{n \times nr}$ matrix has full row rank, that is $rank\,(R) = n$. Moreover, we can verify directly with Matlab through the *ctrb* function.

2.1 Downward position

For this position, we have:

$$R_{down} = \begin{bmatrix} 0 & \frac{4935107427201581}{140737488355328} & -\frac{6218166413817175}{8796093022208} & \frac{6825292479804947}{549755813888} \\ 0 & -\frac{4925410432840623}{140737488355328} & \frac{6378776113532615}{8796093022208} & -\frac{770400999661797}{68719476736} \\ \frac{4935107427201581}{140737488355328} & -\frac{6218166413817175}{8796093022208} & \frac{6825292479804947}{549755813888} & -\frac{3628355444977149}{17179869184} \\ -\frac{4925410432840623}{140737488355328} & \frac{6378776113532615}{8796093022208} & -\frac{770400999661797}{68719476736} & \frac{6328762140136585}{34359738368} \end{bmatrix}$$

and we have $rank\left(R_{down}\right) = 4$, so the system in this configuration is controllable.

2.2 Upward position

In this case, we have:

$$R_{up} = \begin{bmatrix} 0 & \frac{4935107427201581}{140737488355328} & -\frac{6218166413817175}{8796093022208} & \frac{8858444698715643}{549755813888} \\ 0 & \frac{4925410432840623}{140737488355328} & -\frac{6378776113532615}{8796093022208} & \frac{4968681040599551}{274877906944} \\ \frac{4935107427201581}{140737488355328} & -\frac{6218166413817175}{8796093022208} & \frac{8858444698715643}{549755813888} & -\frac{3130720068146841}{8589934592} \\ \frac{4925410432840623}{140737488355328} & -\frac{6378776113532615}{8796093022208} & \frac{4968681040599551}{274877906944} & -\frac{1747114645631177}{4294967296} \end{bmatrix}$$

and again, we have $rank\left(R_{up}\right) = 4$, thus allowing the possibility of designing a control for the system.

Part V
Controller design and implementation

1 Pole placement

Once ensured that (A_{down}, B_{down}) and (A_{up}, B_{up}) pairs are controllable, it is legit to move the poles of the open loop system, such that, once we close the loop, we can obtain the desired behaviour. In particular, from a theoretical point of view, given the linear system $\dot{x} = Ax + Bu$, we can consider the static feedback law $u = K\left(x_{des} - x\right)$,so that considering the closed loop dynamics that are given by $\dot{x} = \left(A - BK\right)x + BKx_{des}$, we converge to the desired input (state) reference x_{des}, with K chosen so that the eigenvalues of $(A - BK)$ are the one desired.

In order to achieve this, we can use several methods such as the Achermann's formula or the Simon-Mitter algorithm (since, as we'll see later, it is enough to move just one and two poles to improve the behaviour of the system). Executively, we will use the *place* function in Matlab.

With a system that is non-canonical (higher order system) the relationships are not exact. One idea is to place one or two dominant poles to meet our requirements and place the rest of poles to be significantly faster, so that we can one sense neglect the effect of the dynamics for the other transient response.

1.1 Downward position

Analising the system, the approximated eigenvalues (used here due to their better readability) are:

$$eigs(A_d) = \left\{ \ 0, \quad -17.376, \quad -1.673 + j6.936, \quad -1.673 - j6.936 \ \right\}$$

Now, we have to find K_d such that the matrix $(A_d - B_d K_d)$ will be asymptotically stable; looking at the eigs, we notice two main things:

1. we have a pole in zero, hence the system is marginaly stable; to solve this problem, it is enough moving it from the origin into the strictly negative real line.
2. the system is not in a canonical form for the second order analysis; however, since the 1) point above, and the fact that the second pole is an order of magnitude faster than the later two (WRONG; see the module), we can see that the behaviour is the one of an underdamped system $\zeta \in (0, 1) = 0.2344$, so the system converges but with a lot of damping; to prevent this, it is enough to get four distinct and asymptotically stable poles, so that the damping factor would be equal to 1, that is the case of *critically damped* systems. (we can verify this with the *damp* function of Matlab)

Thanks to this considerations, we can put as desired eigenvalues:

$$eigs(A_d - B_d K_d) = \left\{ \ -3, \quad -18, \quad -8, \quad -10 \ \right\}$$

where the control law is given by the feedback loop with $u = K_d \left(x_{des} - x \right)$, and:

$$\begin{aligned} K_d &= \left[-\frac{6117366928380881}{2251799813685248} \quad \frac{2971164168543337}{140737488355328} \quad -\frac{1270943529509367}{562949953421312} \quad \frac{97966944119591}{35184372088832} \right] \\ &\approx \left[-2.7167 \quad 21.1114 \quad -2.2576 \quad 2.7844 \right] \end{aligned}$$

1.2 Upward position

In this case, the approximated eigenvalues are:

$$eigs(A_u) = \left\{ \ 0, \quad -22.840, \quad 7.372, \quad -5.253 \ \right\}$$

By request, we want to stabilize the system, and in order to do this, we can make again some considerations:

1. To make the system stable, we have to move the unstable pole so that it will have $Re\left\{\lambda_i\right\} < 0$.
2. We have again a pole in zero, that gives a *marginal stability* of the system. For this reason, we have to move it away from the origin, and make it asymptotically stable.

Again, to improve the readability of the system, we make the poles as integers, so the closed loop system becomes:

$$eigs(A_u - B_u K_u) = \left\{ \ -3, \quad -23, \quad -4, \quad -6 \ \right\}$$

and the matrix of gain used by the input is:

$$\begin{aligned} K_u &= \left[-\frac{4689981311758687}{4503599627370496} \quad \frac{4054537097117843}{281474976710656} \quad -\frac{3142386730601753}{2251799813685248} \quad \frac{2065823870463971}{1125899906842624} \right] \\ &\approx \left[-1.0414 \quad 14.4046 \quad -1.3955 \quad 1.8348 \right] \end{aligned}$$

2 LQR control law

The Linear Quadratic Regulator controller follow the same idea of pole placement, but instead of empirically assign poles in order to satisfy certain requirements, other considerations are made in order to improve its efficiency as controller.

In fact, given the system $\dot{x} = Ax + Bu$, and the state-feedback law $u = -Kx$, we choose the matrix K such that it minimize the cost function:

$$J(u) = \int_0^\infty \left(x^T Q x + u^T R u + 2x^T N u\right) dt$$

In order to minimize it, we have to set $K = R^{-1}\left(B^T P + N^T\right)$, where P is a symmetric matrix that is found solving the continuos time *algebraic Riccati equation* (**ARE**):

$$A^T P + PA - (PB + N) R^{-1}\left(B^T P + N^T\right) + Q = 0$$

while A, B, N, Q, R are known real coefficient matrices (N will be set equal to zero).

2.1 Downward position

By requirement, the matrices must assume the values:

$$Q_d = \begin{bmatrix} 0.01 & 0_{1\times 3} \\ 0_{3\times 1} & I_{3\times 3} \end{bmatrix} \quad R_d = 10$$

Using the *lqr* function in Matlab, we get the matrix K that satisfy all the required conditions presented above. For the sake of completeness, we report also the eigenvalues given as output of the function, that are the one of the closed loop matrix $A_{d_{cl}} = (A_d - B_d K_{lqr})$:

$$\begin{aligned} K_{d_{lqr}} &= \begin{bmatrix} \frac{2278661198716255}{72057594037927936} & -\frac{7428399756562269}{9007199254740992} & \frac{6307372313118589}{72057594037927936} & -\frac{3478848771555317}{36028797018963968} \end{bmatrix} \\ &\approx \begin{bmatrix} 0.0316 & -0.8247 & 0.0875 & -0.0966 \end{bmatrix} \end{aligned}$$

$$eigs(A_{d_{cl}}) = \begin{Bmatrix} -\frac{890264704655807}{18014398509481984} \\ -\frac{4659624664606745}{2251799813685248} + j\frac{7120251430763241}{1125899906842624} \\ -\frac{4659624664606745}{2251799813685248} - j\frac{7120251430763241}{1125899906842624} \\ -\frac{3234408201994563}{140737488355328} \end{Bmatrix} \approx \begin{Bmatrix} -0.0494 \\ -2.0693 + j6.3241 \\ -2.0693 - j6.3241 \\ -22.9819 \end{Bmatrix}$$

From the result found above, we see that instead of using our approach for let the system critically damped, the minimum energy one move the poles just enough to ensure stability, while minimizing the cost.

2.2 Upward position

In this case the matrices involved in the minimization are:

$$Q_u = \begin{bmatrix} 0.01 & 0_{1\times 3} \\ 0_{3\times 1} & I_{3\times 3} \end{bmatrix} \quad R_u = 100$$

while the gain matrix K_u and the eigenvalues become:

$$
\begin{aligned}
K_{u_{lqr}} &= \begin{bmatrix} -\frac{1}{100} & \frac{3554156018742015}{281474976710656} & -\frac{1266987890578365}{1125899906842624} & \frac{1759550344771959}{1125899906842624} \end{bmatrix} \\
&\approx \begin{bmatrix} -0.0100 & 12.6269 & -1.1253 & 1.5628 \end{bmatrix}
\end{aligned}
$$

$$
eigs\left(A_{u_{cl}}\right) = \left\{ \begin{array}{c} -\frac{5099557538736035}{2882303761517117744} \\ -\frac{752962117144551}{140737488355328} \\ -\frac{8079440650005905}{1125899906842624} \\ -\frac{6589480795020015}{281474976710656} \end{array} \right\} \approx \left\{ \begin{array}{c} -0.0177 \\ -5.3501 \\ -7.1760 \\ -23.4105 \end{array} \right\}
$$

Again the poles have been slightly moved, while the unstable pole was almost mirrored in the stable part of the complex plane.

Part VI
Controllers simulation and testing

Once done every theoretical analysis, we can proceed into simulation. To make the things better, we can make a Simulink model in which we can highlight both structure and workflow of the solution.

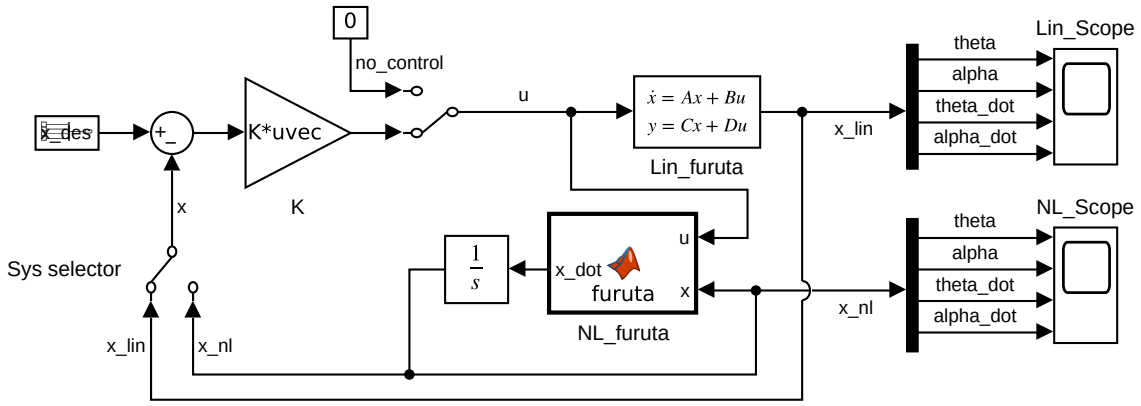

Figure 1: Simulink Furuta model.

Where in the linear system the matrices for the output part are defined as $C = eye\left(4\right)$ and $D = zeros\left(4, 1\right)$ in order to maintain the state unchanged. Moreover, the linear model will see its matrices changed each time we want to test a different configuration, the non linear system is given by the Matlab function below:

```
function x_dot = furuta(u,x)
% Furuta pendulum

% renaming state in compliance with altready written code

% th = x(1); % not used variable, saving calculations
a = x(2); % Downward position
% a=x(2)+pi; % Upward position
th_d = x(3);
a_d = x(4);

% assigning values to symbolic values

Rm = 2.6; Kt = 7.68e-3; Em = 0.69; Km = 7.68e-3; Kg = 70;
Eg = 0.9; mp = 0.127; lp = 0.1556; Jp = 0.0012; Jarm = 0.002;
Bp = 0.0024; Barm = 0.0024; r = 0.2159; g = 9.81;

% defining matrices
D = [(r^2)*mp+(lp^2)*mp-(lp^2)*(cos(a)^2)*mp+Jarm, r*cos(a)*mp*lp;
    r*cos(a)*mp*lp                                , (lp^2)*mp+Jp];

C = [2*mp*cos(a)*a_d*(lp^2)*sin(a), -mp*sin(a)*a_d*lp*r;
    -mp*cos(a)*th_d*(lp^2)*sin(a) ,            0        ];

G = [     0;
    mp*g*sin(a)*lp];

tau_m = Eg*Kg*Em*Kt*(u-Kg*Km*th_d)/Rm;

tau = [tau_m - Barm*th_d;
            -Bp*a_d     ];

% defining the non-linear equations
x_dot  = zeros(4,1);

x_dot(1:2) = [th_d;
              a_d];
x_dot(3:4) = (D^-1)*(tau - G - C*[th_d; a_d]);

end
```

Notice that the nonlinear system is written so that the state space is in accordance with the downward position; for this reason, when we want to run the simulation in the upward one, we have to offset the state so that we can consider the upward position as origin for both the linearized and the nonlinear system, we have to offset it of $-\pi$ for the α part.

1 Open loop response

As per the specification, we want to observe the behaviour of the linearized system when no input is applied (the nonlinear simulations will be done later in the coursework, at the section 5).

As initial condition, the following vector has been chosen:

$$\begin{bmatrix} x_1(0) & x_2(0) & x_3(0) & x_4(0) \end{bmatrix}^T = \begin{bmatrix} 0.5 & \pi/4 & 0.3 & 0.1 \end{bmatrix}^T$$

This initial condition has been chosen so that the linearized system follow quite well the non linear one, but the differences in their evolutions start to show up.

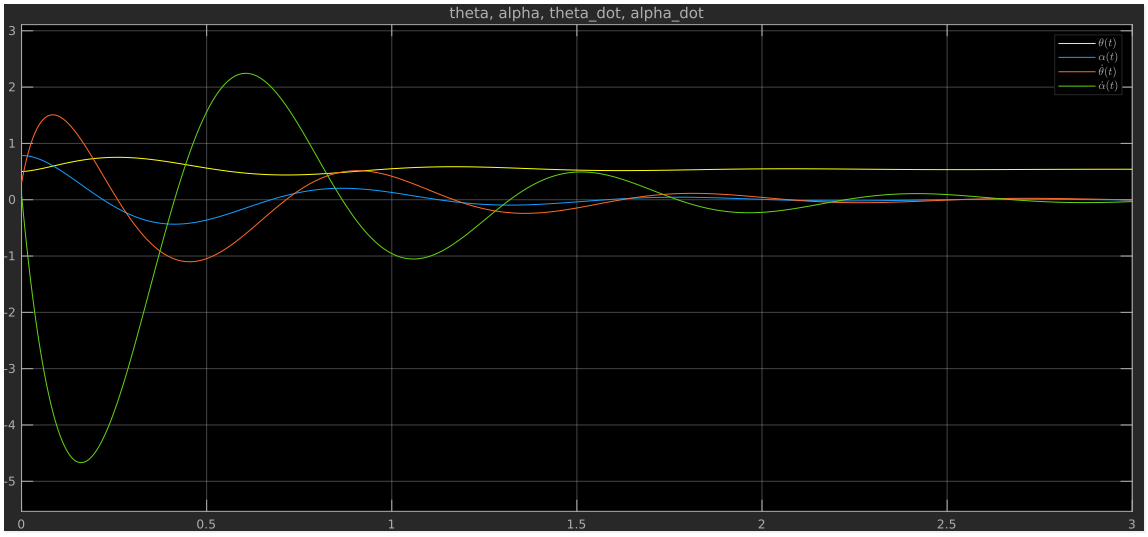

Figure 2: Open loop response of the linearized system.

We can then zoom over the α variable (the one with highest interest), in order to show its trajectory over time. As we can notice, the underdamped behaviour is noticeable, so that the system goes to zero like oscillating around zero.

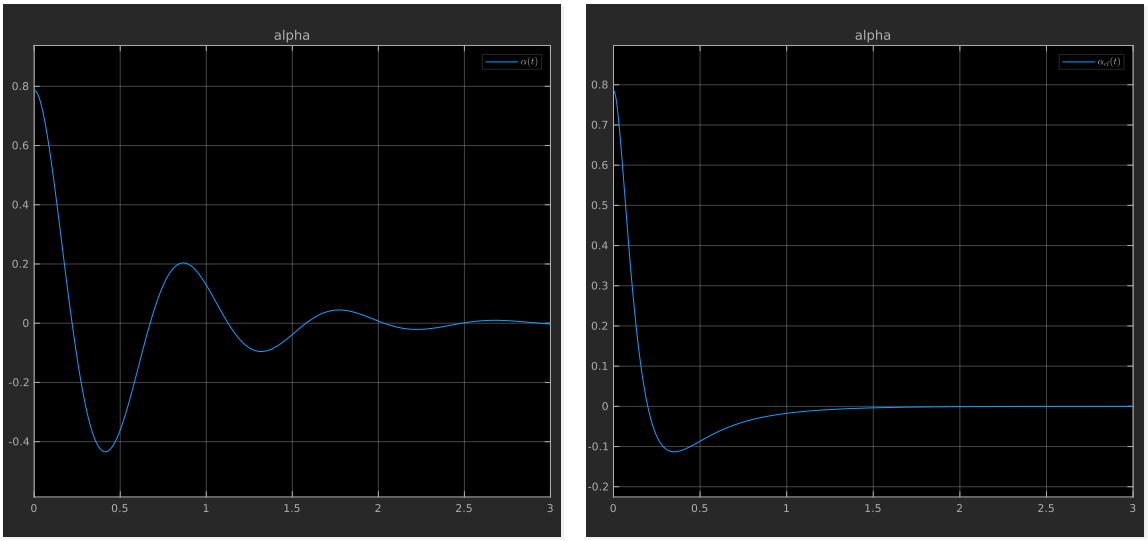

Figure 3: Evolution over time for $\alpha(t)$ and $\alpha_{cl}(t)$ (without and with control).

As we can see from figures above, the system controlled with $K_{d,pp}$ control now have a very slight overshoot, while the damping is now gone (we converge from the same side to zero, instead of oscillating around it).

2 Square-wave reference

As request, we have now to feed a desired reference that changes over time; we can expect two different behaviours:

- The frequency of the square wave is *low* w.r.t the dynamics of the system; in this case, the controlled pendulum will reach the target state (asymptotically) before moving again once the set-point is changed;
- The frequency is high; in this case, the physical system will not be sufficiently fast to keep up with the change of the target value, hence the state of the system will oscillate trying to follow the input, but with a lower amplitude. Therefore, we say say that the system acts as a Low Pass Filter (LPF) with respect to the input signal.

In the simulations, the same initial conditions as the previous point are applied (even if after the initial transient behaviour, the system will follow the dynamics imposed by the reference signal, thus making them useless); also, two cases are presented, for low and high frequency.

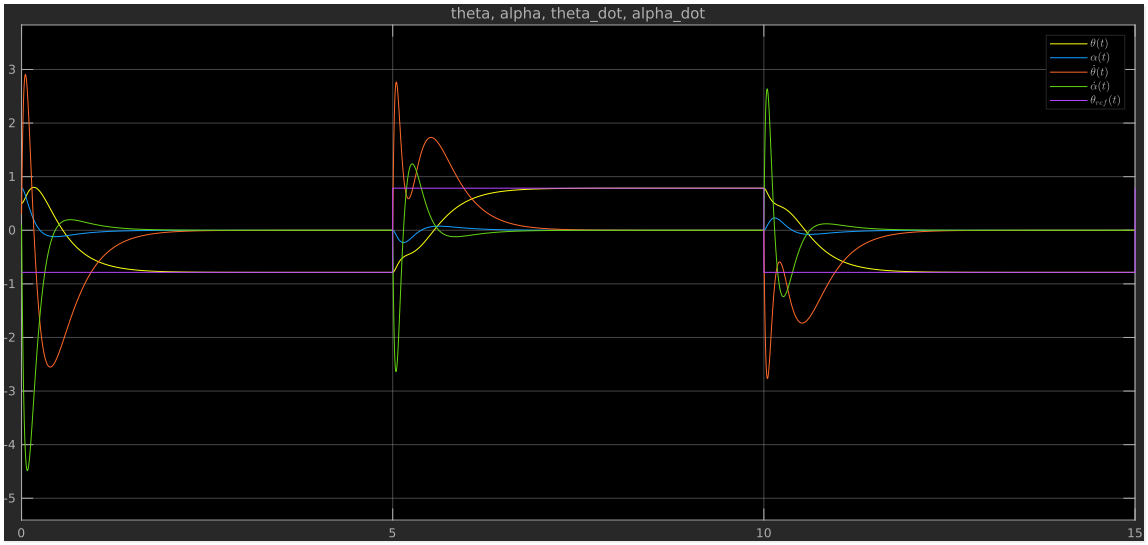

Figure 4: State evolution over time with frequency of $\theta_{ref}\,(t) = 0.1Hz$.

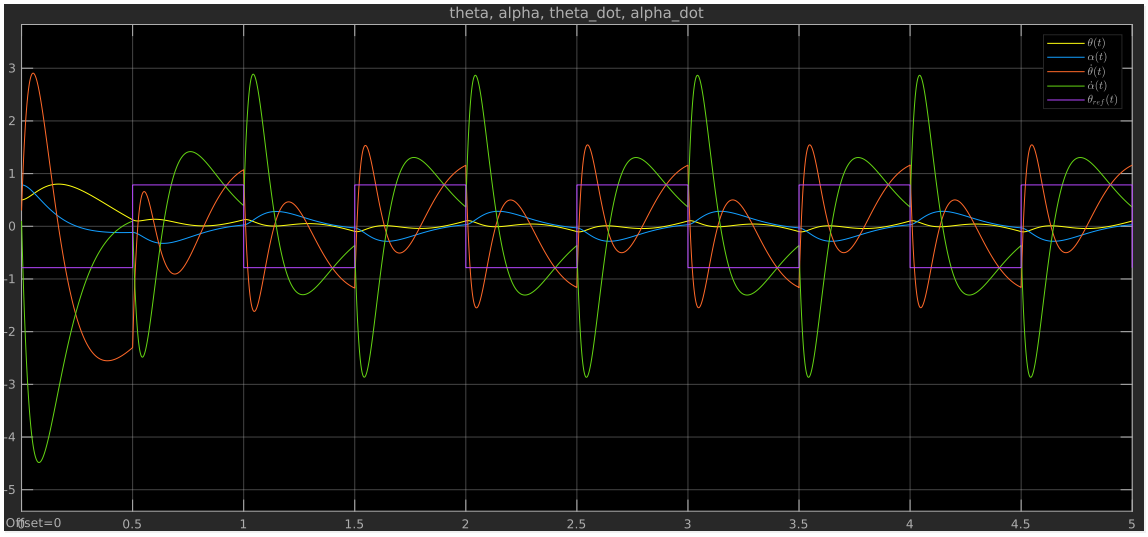

Figure 5: State evolution over time with frequency of $\theta_{ref}\,(t) = 1Hz$.

As we can see from the simulations above, the results adheres to the discussion presented before.

3 LQR performance

As presented before, the LQR is characterized by the attempt to reduce the energy used by control globally. For this reason, it is reasonable expecting that the speed of the controller depends on weights used in Q and R matrices, while the time of convergence will be higher (by also looking at the eigenvalues of the closed loop matrix $A_{cl,lqr}$). In particular, with a very small weight for the $x_1$component of the state, we can expect that the error for $\theta(t)$ would be "tolerated" more, hence putting less effort for it.

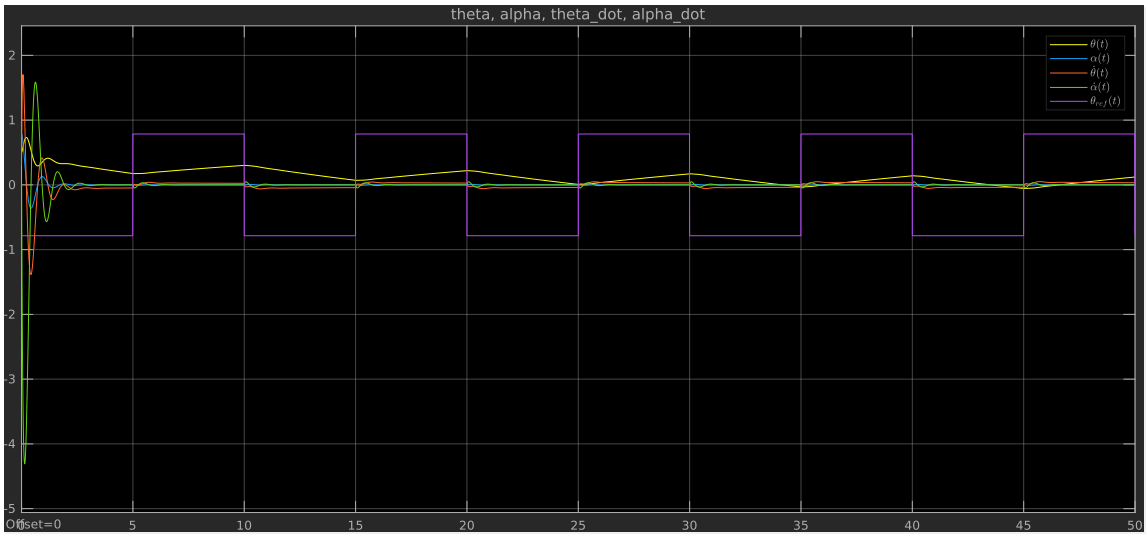

Figure 6: State evolution over time with frequency of $\theta_{ref}(t) = 0.1Hz$.

As previously supposed, $\alpha, \dot{\theta}, \dot{\alpha}$ go to their reference quickly, while θ, having a much lower associated weight, it is not able to reach the set point value that changes over time (it would work for very slow square wave signals).

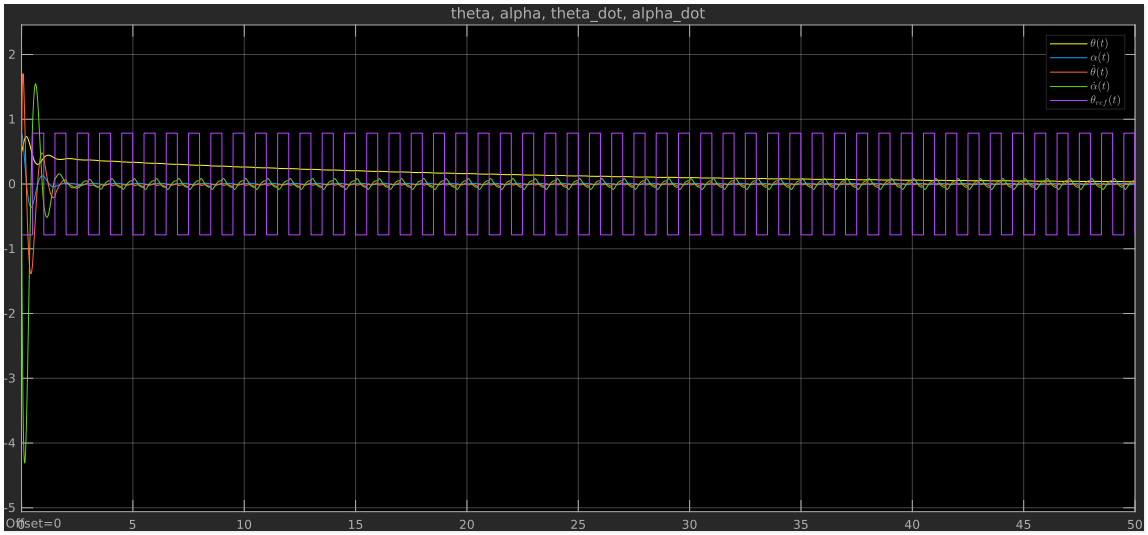

Figure 7: State evolution over time with frequency of $\theta_{ref}\,(t) = 1Hz$.

Increasing the frequency worsens the behaviour, with θ that converges even more slowly and pushing itself nearer to zero.

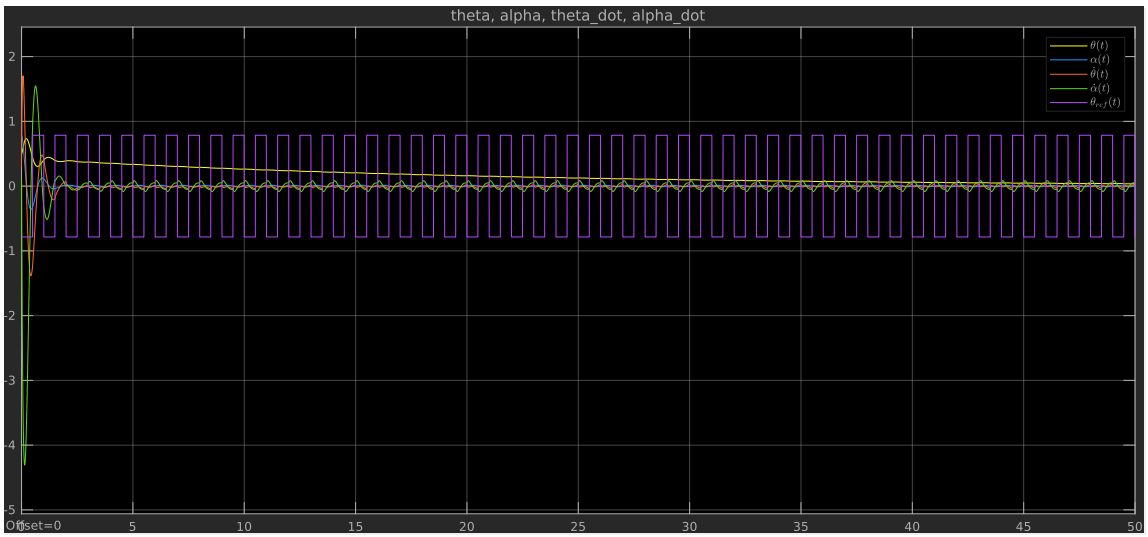

Figure 8: State evolution over time with frequency of $\theta_{ref}\,(t) = 1Hz$.

4 Upward position

For the upright position we have to bear in mind that now the origin of the state space is upwards, so that we can make a linear system as it would be. By requirements, we have to set $\theta(t) = \pi, \forall t \in \mathbb{R}$ for all the following simulations, where we will validate our controller in the linearized case.

4.1 Constant reference signal

For this case, we want to reach the unstable equilibrium in the upward position, starting with similar initial conditions to the ones presented in the previous section:

$$\begin{bmatrix} x_1(0) & x_2(0) & x_3(0) & x_4(0) \end{bmatrix}^T = \begin{bmatrix} 0.5 & \pi/8 & 0.3 & 0.1 \end{bmatrix}^T$$

In this case, we start with an angle of 22.5° from the vertical axis, so we are sufficiently unbalanced to need a real quick control to stabilize the pendulum (NOT WRITE THIS: remember that the energy required and speed of poles approaches infinity for $\alpha \longrightarrow \pm\frac{\pi}{2}$, this without exploiting the whole state space and constraining it to only the upper part for i.e. admissibility reasons).

For instance, the complete reference for the whole state space is:

$$\begin{bmatrix} x_{1,ref}(t) & x_{2,ref}(t) & x_{3,ref}(t) & x_{4,ref}(t) \end{bmatrix}^T = \begin{bmatrix} \pi & 0 & 0 & 0 \end{bmatrix}^T$$

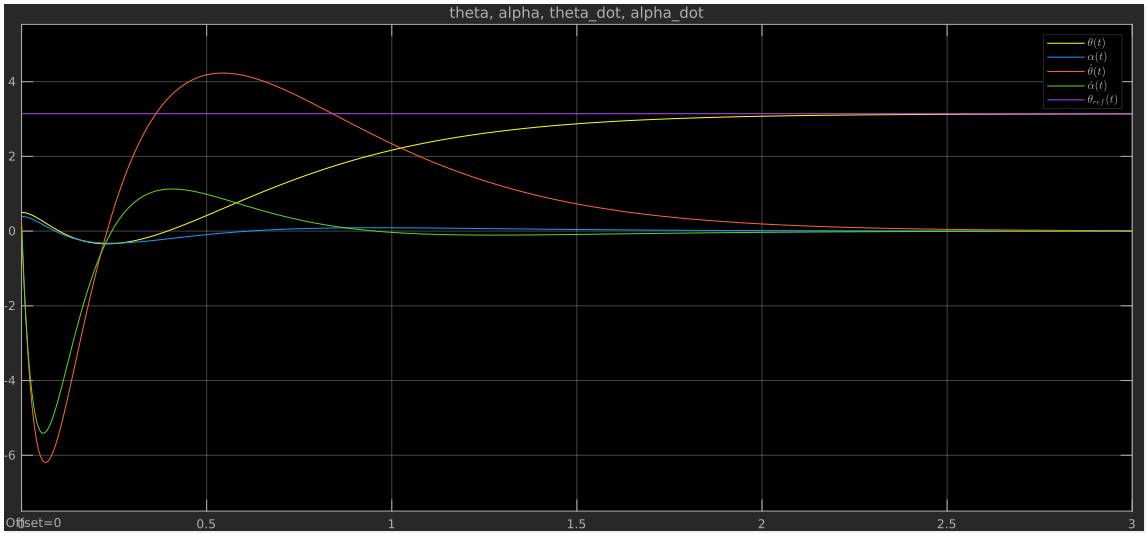

Figure 9: State evolution over time with $K_{up,pp}$ control.

As we can see, the state goes to steady state in few seconds, stabilizing the system to the desired reference.

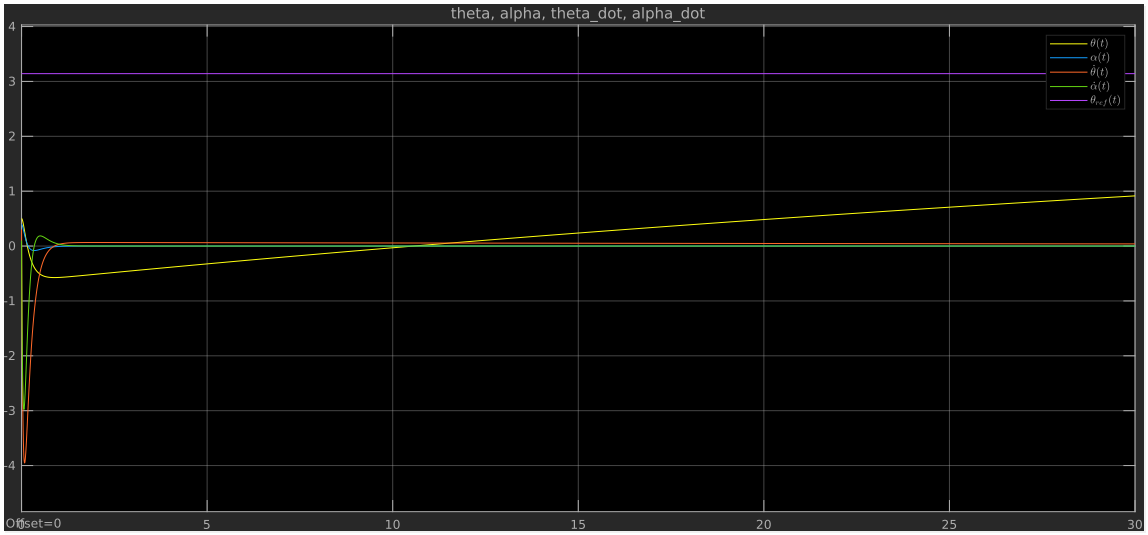

Figure 10: State evolution over time with $K_{up,lqr}$ control.

Similarly to the previous section, the LQR section shows up a behaviour that is similar to the previous one for $\alpha, \dot{\theta}, \dot{\alpha}$, while being two order of magnitude slower for θ (even if not shown, the x_1 state approaches the reference signal at around 300 seconds, 100 times slower with respect to the other controller and in alignment with a 100 times smaller weight for its component in the matrix Q).

4.2 Square-wave reference signal

This time, in order to have less transient and fit better on paper the simulations found, the initial conditions will be:

$$\begin{bmatrix} x_1(0) & x_2(0) & x_3(0) & x_4(0) \end{bmatrix}^T = \begin{bmatrix} \pi - 0.1 & \pi/8 & 0.3 & 0.1 \end{bmatrix}^T$$

In this way, the transient time will be small, and we can easily observe the steady state oscillations (WRONG: since both linearized systems are controllable, we were able to let the eigenvalues to be stable and thus we expect to have similar behaviours for the upward and downward system, without considering the different temporal constants).

Moreover, this time the component of the state that oscillates is the $\alpha(t)$.

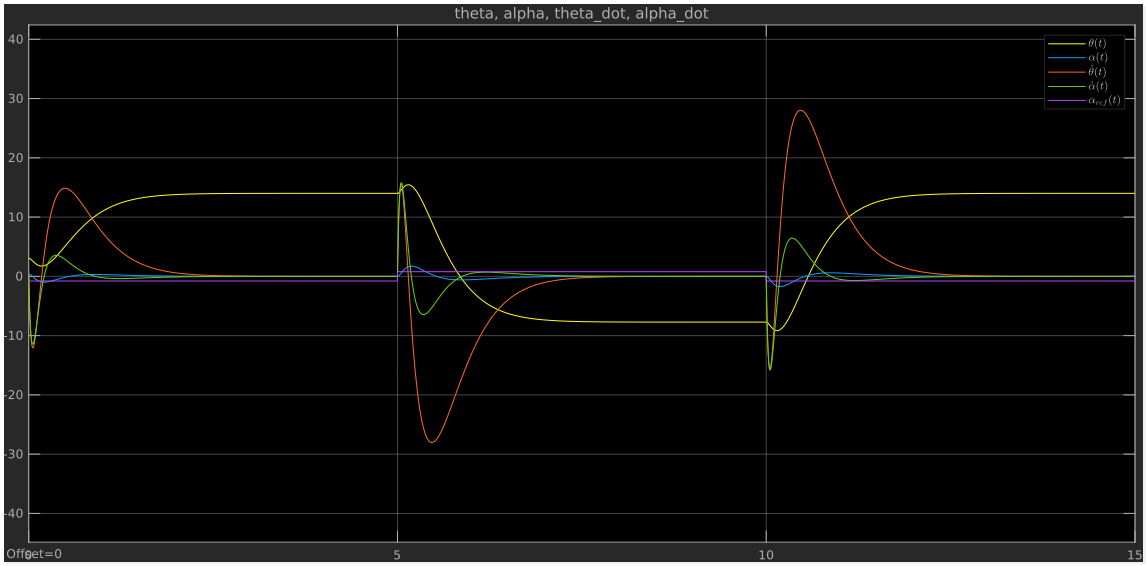

Figure 11: State evolution over time with $K_{up,pp}$ control and $\alpha_{ref}(t) = 0.1Hz$.

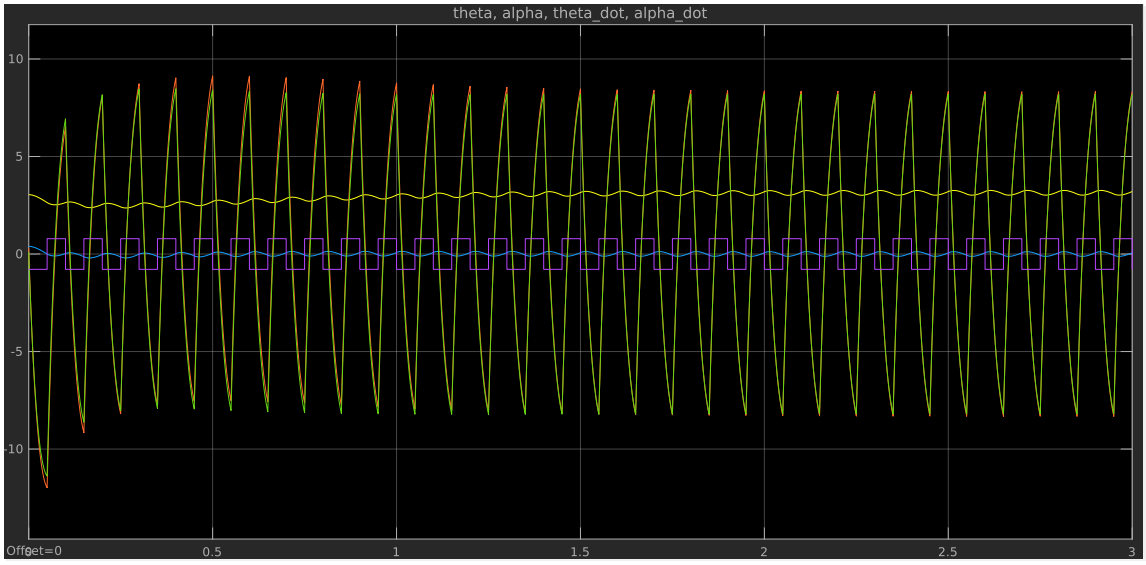

Figure 12: State evolution over time with $K_{up,pp}$ control and $\alpha_{ref}(t) = 10Hz$.

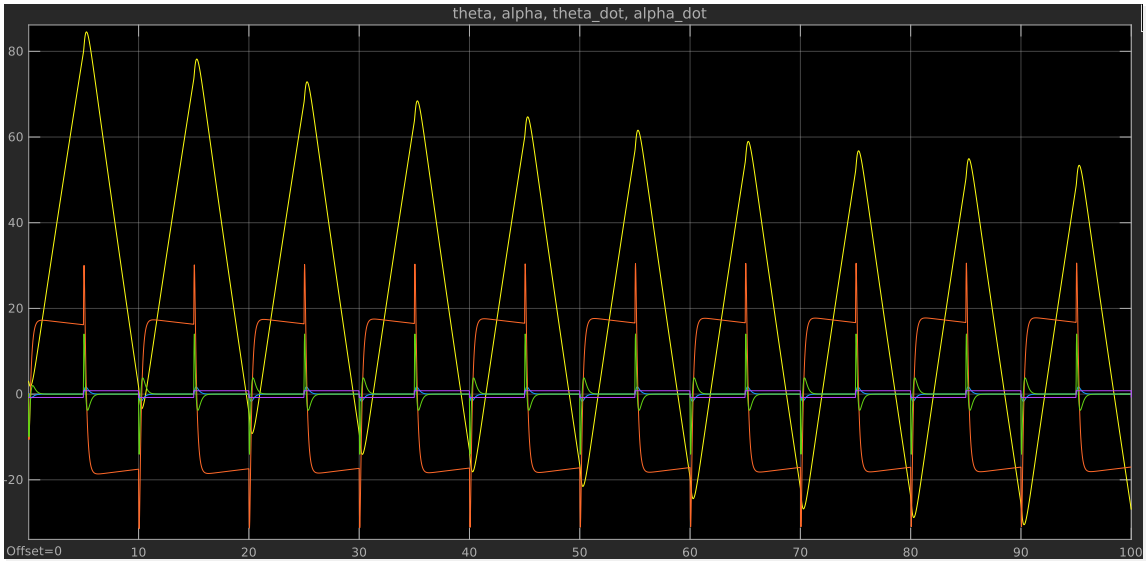

Figure 13: State evolution over time with $K_{up,lqr}$ control and $\alpha_{ref}\left(t\right) = 0.1Hz$.

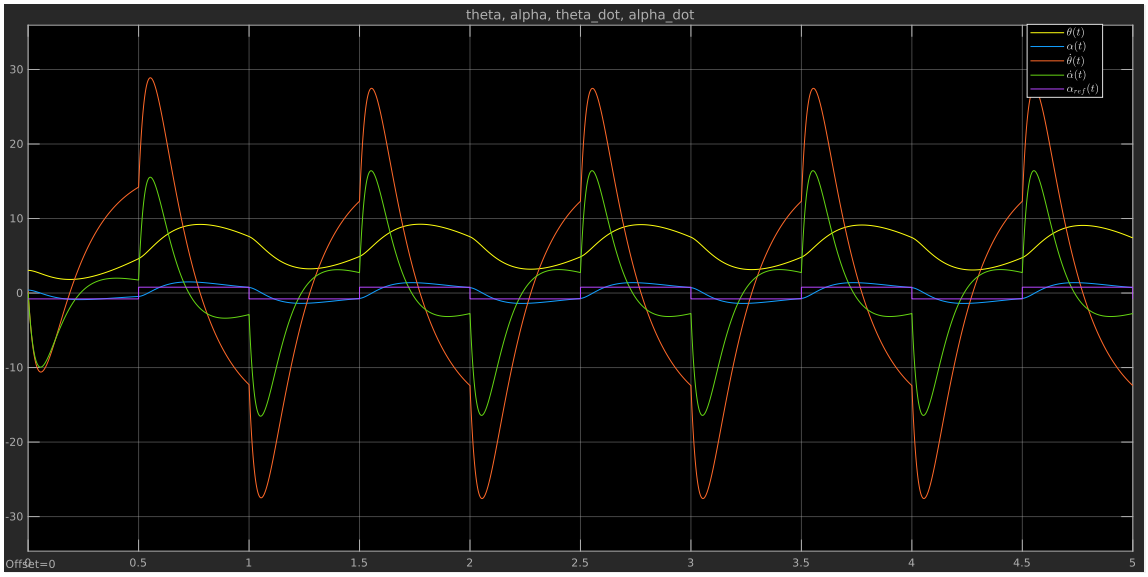

Figure 14: State evolution over time with $K_{up,lqr}$ control and $\alpha_{ref}\left(t\right) = 1Hz$.

10 Hz LQR not done since it does not add anything useful.

NOT SURE: Moreover, we can observe that at 1 Hz we have similar to the natural frequency for the theta and alpha.

We can see that since the reference is set to an unstable angle for the system, it cannot maintain

it; also logically, to maintain a certain angle alpha, we would have to have a theta that increases like a ramp, while the speed thetaDot must be constant. But since the system is asymptotically stable, this behaviour is not allowed (???).

5 Nonlinear system

Throughout this session, the same simulations already seen will be run, to find out that only for a neighbourhood of the origin the linearized and non linear systems will work similarly (approximately few degrees, that means tenths of radians). Both the two cases will be analysed, the downward and the upward position, so we have to change coordinates for simplicity in order to have the same origin stabilization criterion.

5.1 Downward position

5.1.1 Open loop response

Also in the non linear case, it is interesting to see the open loop response; in particular, we expect that the shape of state evolution would be the more similar to the linearized one the less the initial condition differs from the origin. As said in the linearized case above, this imply that the chosen initial conditions

$$\begin{bmatrix} x_1(0) & x_2(0) & x_3(0) & x_4(0) \end{bmatrix}^T = \begin{bmatrix} 0.5 & \pi/4 & 0.3 & 0.1 \end{bmatrix}^T$$

are sufficiently moved from the origin to show a difference between the two behaviours, but not enough to be quite similar.

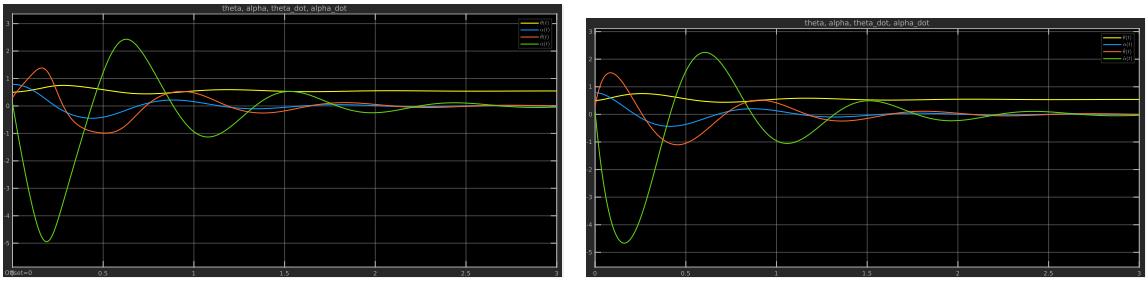

Figure 15: Open loop response of the nonlinear system (left) compared with the linearized (right).

As we can notice from the above plot, $\dot{\theta}$ initially has a sharper shape, while $\dot{\alpha}$ reaches bigger values in module, given by the nonlinear components neglected. Overall, for such small initial conditions, the behaviour is quite similar.

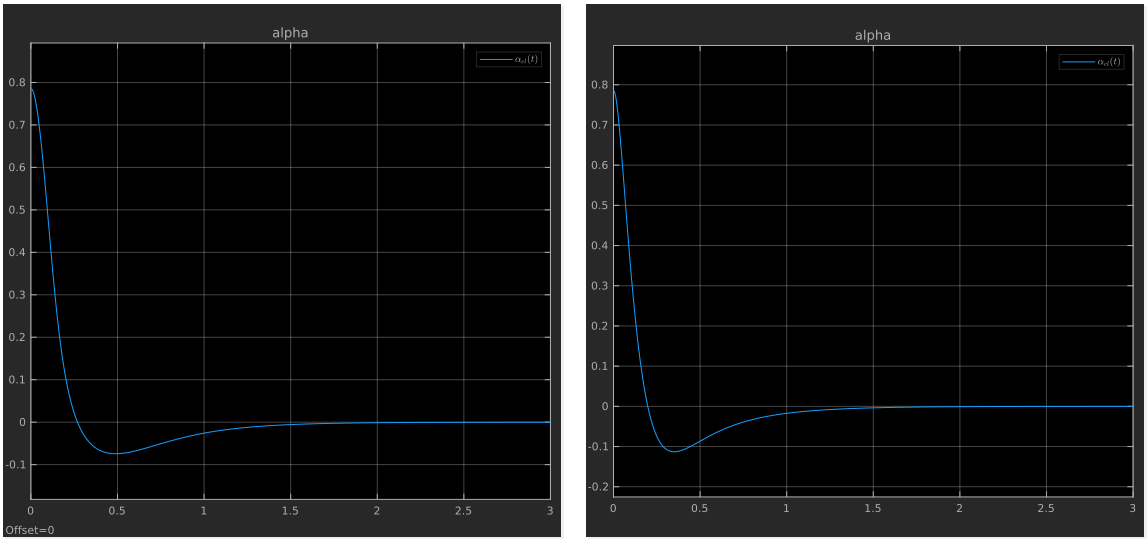

Figure 16: Evolution over time for $\alpha_{cl}(t)$ with $K_{d,pp}$ for NL and Lin systems.

Focusing on the $\alpha(t)$ behaviour, we can observe that it is quite similar in both cases, with slighly less overshoot and slightly longer setting time for nonlinear system. Overall, the initial conditions near zero allow to reduce differences.

5.1.2 Basin of convergence

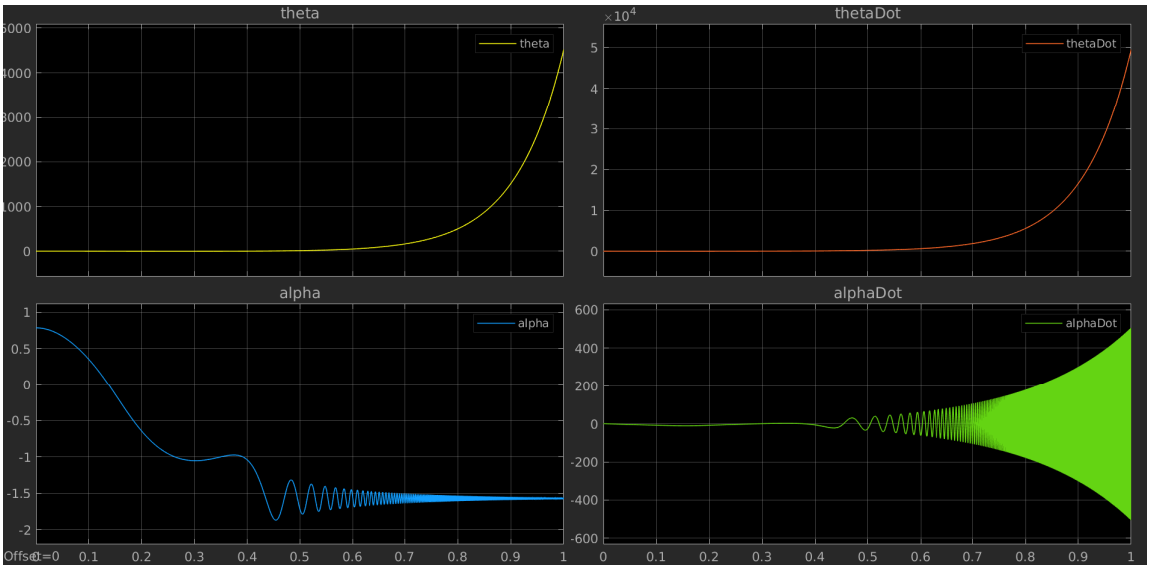

Figure 17: Evolution over time from an unstable condition $\left(K_{d,pp}\ ,\ x_0 = [0, \pi/4, 0, 0]^T\right)$ of the NL system.

Before proceeding in further simulations, it is useful to introduce some boundaries for possible initial conditions. As we know, linearized systems can describe the behaviour of a system only in a neighbourhood of the origin. It follows that it can be defined a specific interval of initial conditions for which the system is stabilized, hence the controller works also for the real case.

From the simulation above, we can see that the control is not enough to bring the state to zero before the gravity pull it to the downward position. At that point the system starts to oscillates around $\frac{\pi}{2}$ while keeping to accelerate along the θ variable, trying to keep the pendulum up. It is obvious that in the real world, we are limited about the speed we can reach, so it is inevitable that the arm will finally go in the downward position.

The simulation when the system is in this case is very slow, but we can exploit this behaviour, as we can see below.

To do so, the system has been ported as function to be integrated through ode45, and the following matlab code has been written:

```
%% Nonlinear evolution (ODE)

% Check stability
steps = 15; % USE AT LEAST TWO.
state_blue = [];
state_red = [];
for alpha = -pi/2:(2*pi/(4*(steps-1))):pi/2
    for theta = -pi:(2*pi/((steps-1))):pi

        x_init = [theta; alpha; 0; 0];
        tstart = tic;
        [~,x] = ode45(@(t,x) furuta_ode(t,x,K_lqr_u),...
                [0,10],x_init,...
                odeset('Events',@(t,x) time_event(t,x,tstart)));

        error = abs( x(end,2)/(2*pi) - round(x(end,2)/(2*pi)) );
        if (error < 0.1)
            state_blue(end+1,:) = [theta, alpha];
        else
            state_red(end+1,:) = [theta, alpha];
        end
    end
end
```

In this way, we can test many initial conditions at once, finding the limit for which the system succeed at stabilize itself. However, some problems arises if we go in blindly:

- If the system is unstable, the simulation goes really slow, since Matlab is trying to integrate a lot of point that are diverging. To avoid inhuman times, we can bound our simulation so that it cannot lasts more than 2 seconds of real time; in fact, all the simulations end up in tenths of seconds when they converge, and if not, cutting the simulation and seeing that the error is still relevant (if $|\alpha(t)| \approx \frac{\pi}{2}$, the variable *error* will be around 0.5), means that the system has not been converged, and it will never do.
- The system cannot be stabilized for sure for $|\alpha(0)| \geq \frac{\pi}{2}$ (at least with simpler controller, that does not exploit the swinging from the downward position); for this reason, we can reduce the interval upon which we select initial conditions.
- $\theta \in [-\pi, \pi]$; moreover, thinking about the physical system, its initial condition marginally influences the ability of converging. In fact, very big initial conditions could influence the stability properties, however, it is enough to set some slow asymptotically poles such that its initial convergence can be sacrificed to let α to reach the upward position, and then converging to the global origin. As this approach suggests, it is the same that justify the use of LQR controller, such that we can conveniently choose weights to be assigned accordingly to a certain priority given by us.

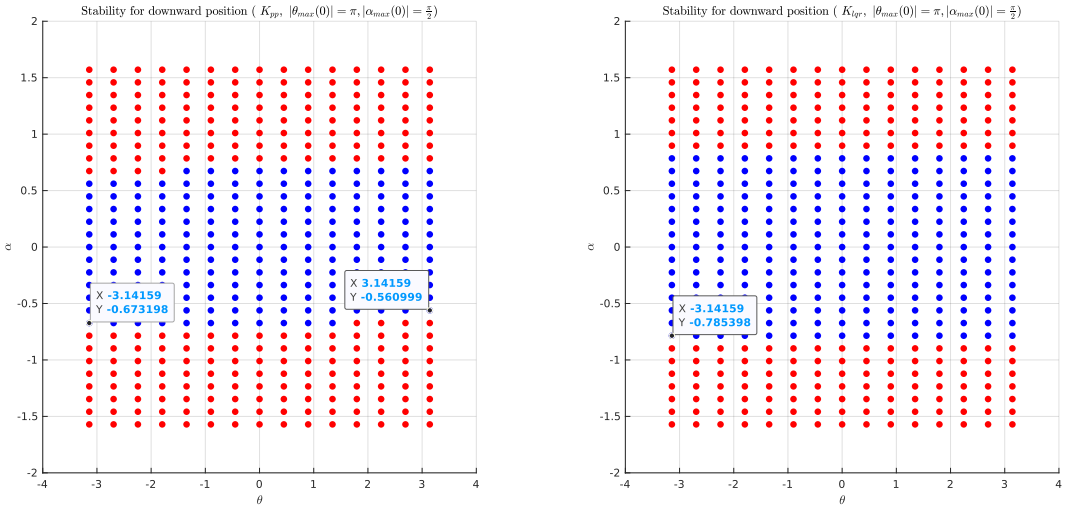

Figure 18: Upward (typo in figure) stability of initial conditions with $K_{d,pp}$ and $K_{d,lqr}$ controller.

As previously thought, the system converges for only a neighbourhood of the origin; moreover, the LQR approach proves to be better since does not force θ to converge too fast, leading to a larger band of controllability. Another detail is that the symmetry of the maximum controllable α angle has a diagonal simmetry, not vertical. This is expected because the pendulum has to rapidly increase θ to keep up with its falling upper part, so for each α_0, there will be for sure an optimal starting value $\theta_0^* \in \mathbb{R}$ such that following a certain profile $\theta^*(t)$, the system will be stabilized. The problem is that we cannot find an easy closed formula to determine it, and moreover every 2π the system reaches the target value for θ, so we have a limited space to build speed and we have to rely on faster poles, that may break the stability.

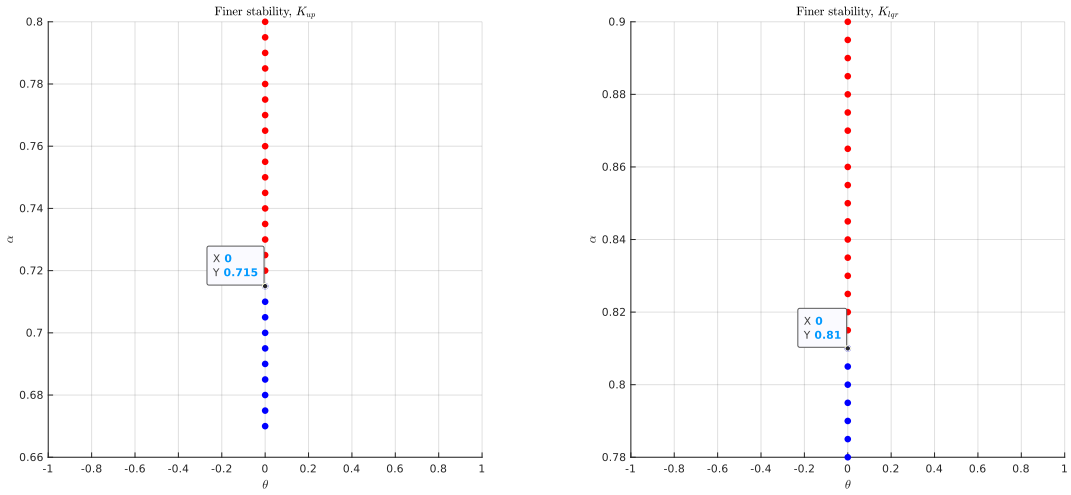

Figure 19: Finer upward stability of at $\theta_0 = 0$, with $K_{d,pp}$ and $K_{d,lqr}$ controller.

Due to the long time required for simulating each initial condition, we stick with a specific θ_0 and then span the interval between the greatest stable $x_0$and the smallest unstable one, from the previous simulation, increasing the number of points between the above-mentioned conditions. As we can notice from the figure, where the last stable initial condition is highlighted, the LQR control has an advantage over the arbitrary pole placement approach, that increases as much as we move away from the $\theta_0 = 0$ condition. For instance, with this θ_0, we can stabilize the system until at least $40.9665°, K_{pp}$, or $46.4096°, K_{lqr}$; as results, the next simulations will be all with an angle $|\alpha_0| \leq \frac{\pi}{6}$.

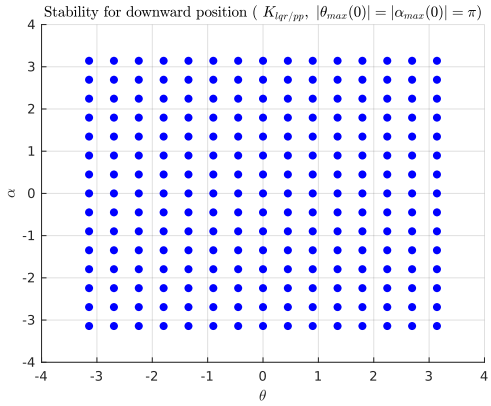

Figure 20: Downward stability of the NL system.

As can be imagined, the downward position does not have any problem to be reached by any controller.

5.1.3 Constant reference signal

We are now ready to proceeding to remaining simulations. First of all, we want to see what happens when we try to follow a constant reference signal, both up and downwards. To have a credible interval, we set a $\alpha = \pi/8$, $\theta = \pi$ reference both downward and upward position, while starting from the two origins (we refer in this way to the two equilibria).

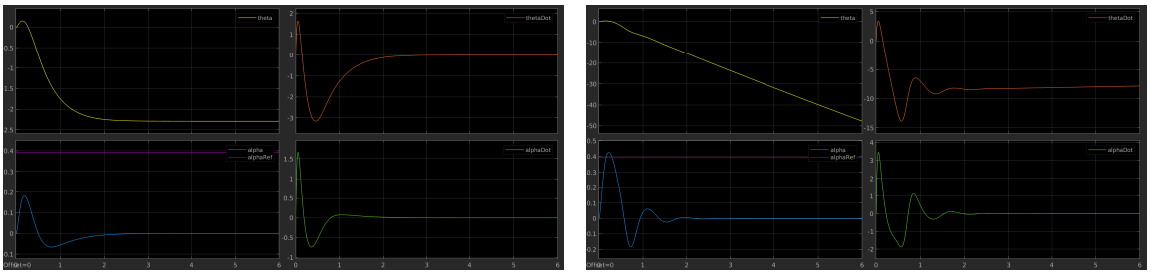

Figure 21: Upward stability, with $K_{u,pp}$ and $K_{u,lqr}$ controller of the NL system.

As we can see from the figure, the system try to follow the reference signal (more in the case of the LQR control); however, it is impossible to keep it constantly (we ought to have a θ that always increases as well as the speed), so α goes eventually to zero. The main difference between the two controller is that thanks to the lower weight over the x_1 component for the LQR controller, the θ angle can go further, doing more spins before settling (as we can see from the decreasing value of $\dot{\theta}$) over some $(2k+1)\,\pi$ position, plus an error given by a non zero reference for α.

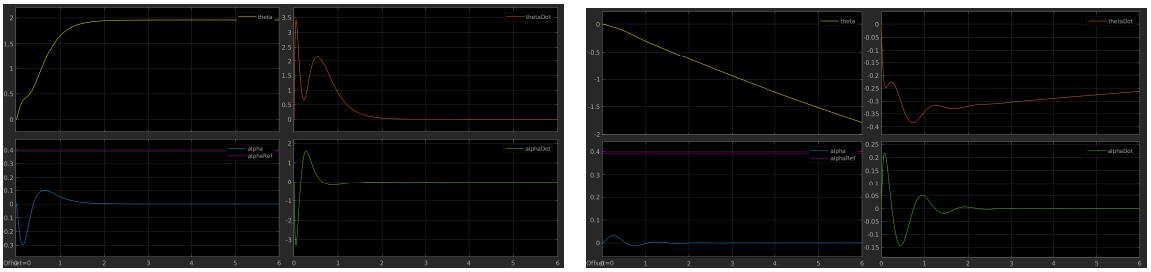

Figure 22: Downward stability, with $K_{d,pp}$ and $K_{d,lqr}$ controller of the NL system.

In this case we have a very similar behaviour, with obviously lower values for θ since the stable behaviour of the downward position and a lower cost in the control part of the cost function (remember that $R_{down} = 10$).

5.2 Square-wave reference

For this last section, we have to repeat the tracking experiment done previously; for this reason, we will have two subsection, one for the downward position and one for the upward

5.2.1 Downward position

In this case, α is set to a constant 0 reference signal, while the θ angle oscillates. As the previous case, we will have both high and low frequency, as well as different amplitudes.

Low Frequency (f=0.1 Hz) As

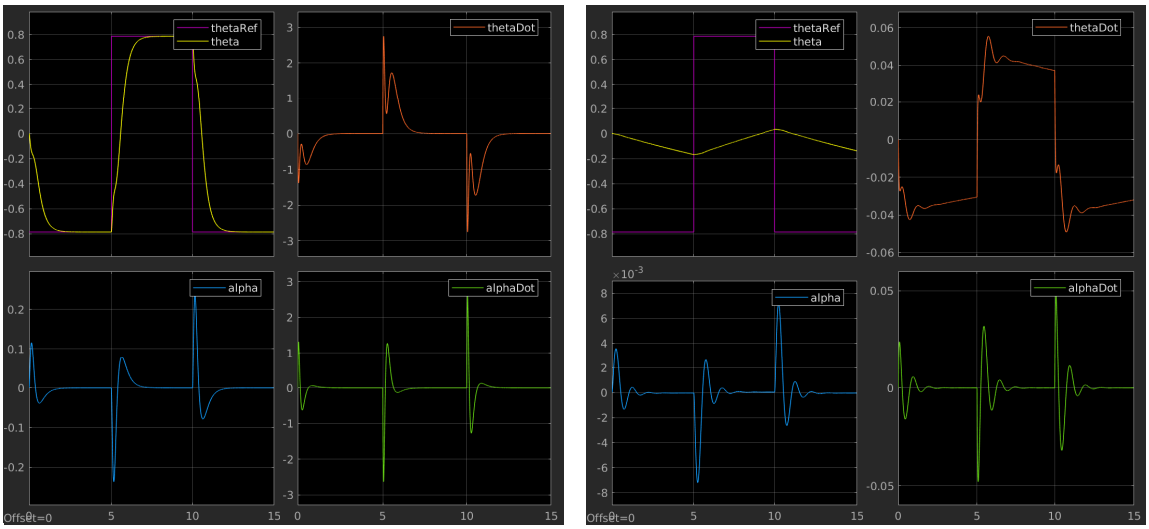

Figure 23: State evolution over time with $K_{d,pp}$ and $K_{d,lqr}$ controller, $\theta_{ref} = \pm 45°$.

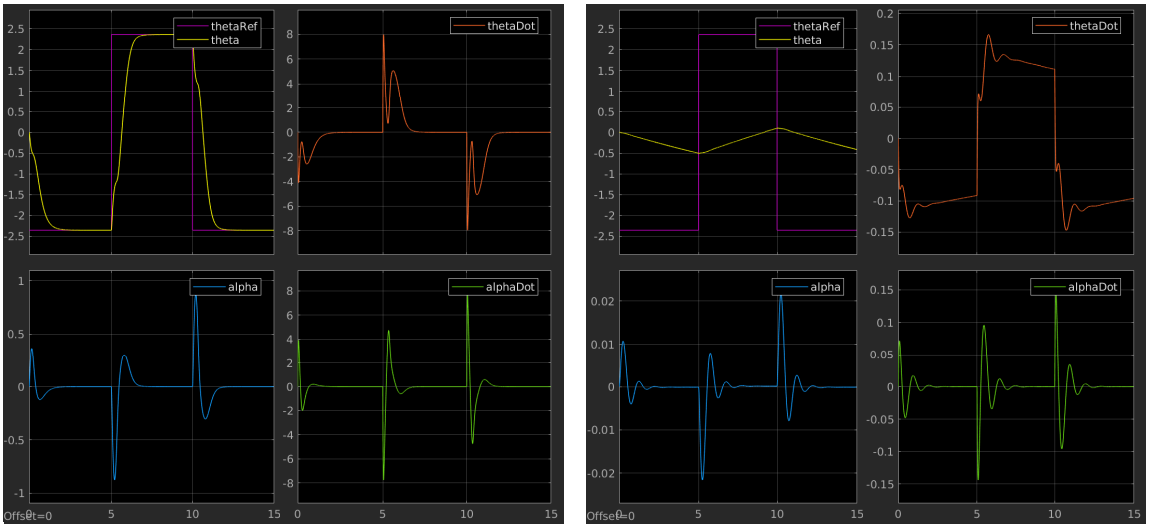

Figure 24: State evolution over time with $K_{d,pp}$ and $K_{d,lqr}$ controller, $\theta_{ref} = \pm 135°$.

High Frequency (f=1 Hz) As

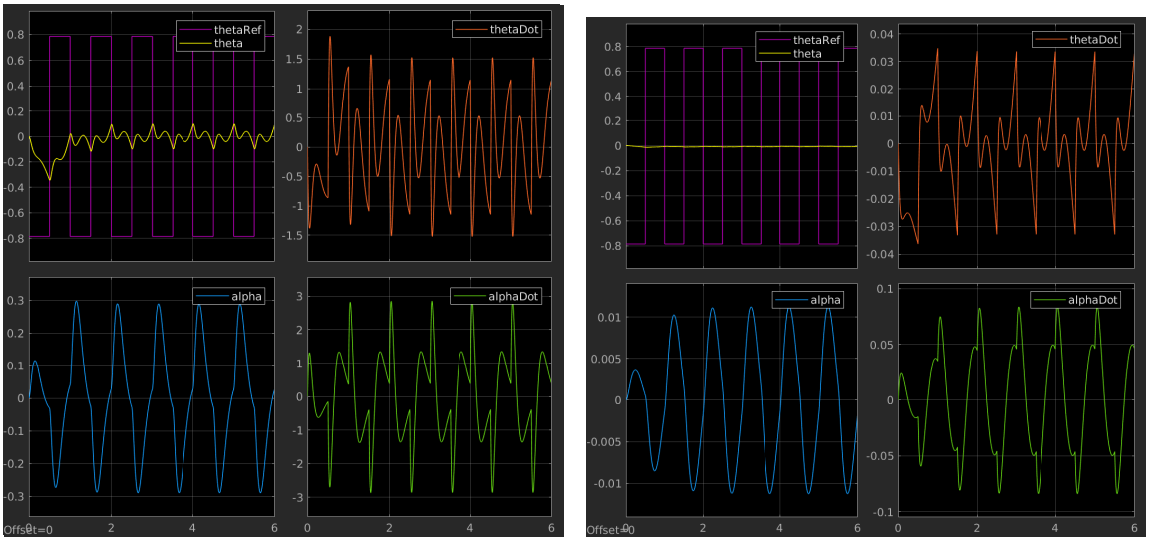

Figure 25: State evolution over time with $K_{d,pp}$ and $K_{d,lqr}$ controller, $\theta_{ref} = \pm 45°$.

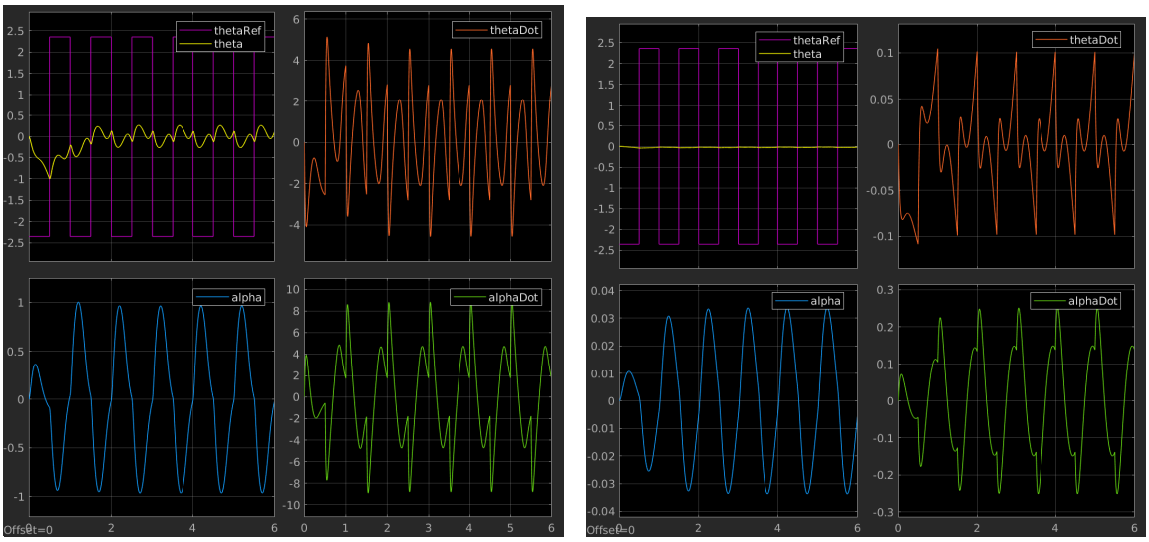

Figure 26: State evolution over time with $K_{d,pp}$ and $K_{d,lqr}$ controller, $\theta_{ref} = \pm 135°$.

5.2.2 Upward position

This time, we have two degrees of freedom along which we can increase the amplitude; in fact, in addiction to α_0 as analysed before, for which after a certain amplitude the system is unstable, if

the control is too aggressive along θ we may end up in breaking stability, as we will later see.

For the α condition, we have chosen one that accordingly to the stability simulations is stable for both, and one that it is not.

Again, the initial condition will be the upward origin for each simulation.

Low Frequency (f=0.1 Hz), θ changes. As

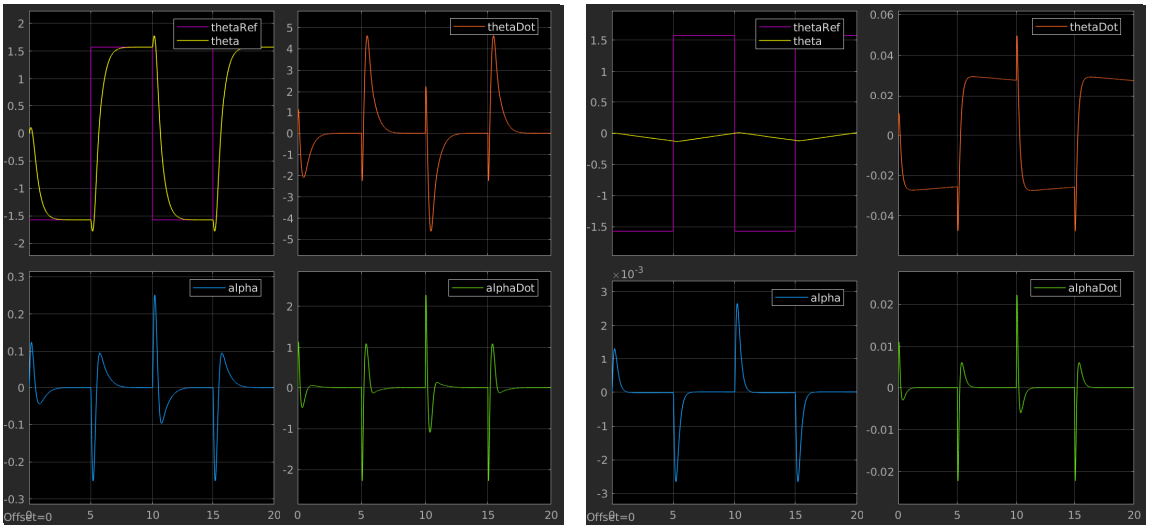

Figure 27: State evolution over time with $K_{u,pp}$ and $K_{u,lqr}$ controller, $\theta_{ref} = \pm 90°$.

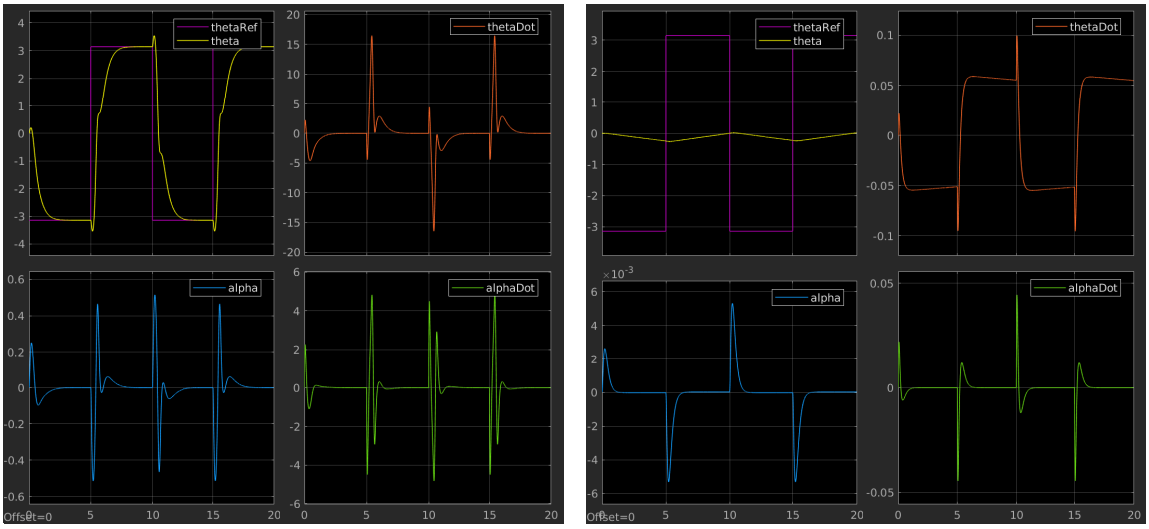

Figure 28: State evolution over time with $K_{u,pp}$ and $K_{u,lqr}$ controller, $\theta_{ref} = \pm 180°$.

High Frequency (f=1 Hz), θ changes. As

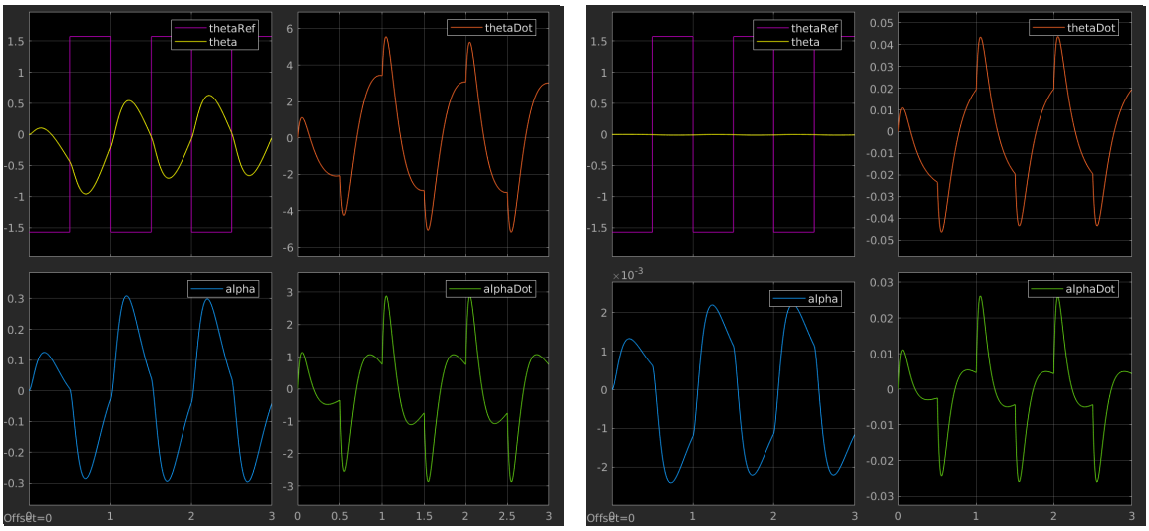

Figure 29: State evolution over time with $K_{u,pp}$ and $K_{u,lqr}$ controller, $\theta_{ref} = \pm 90°$.

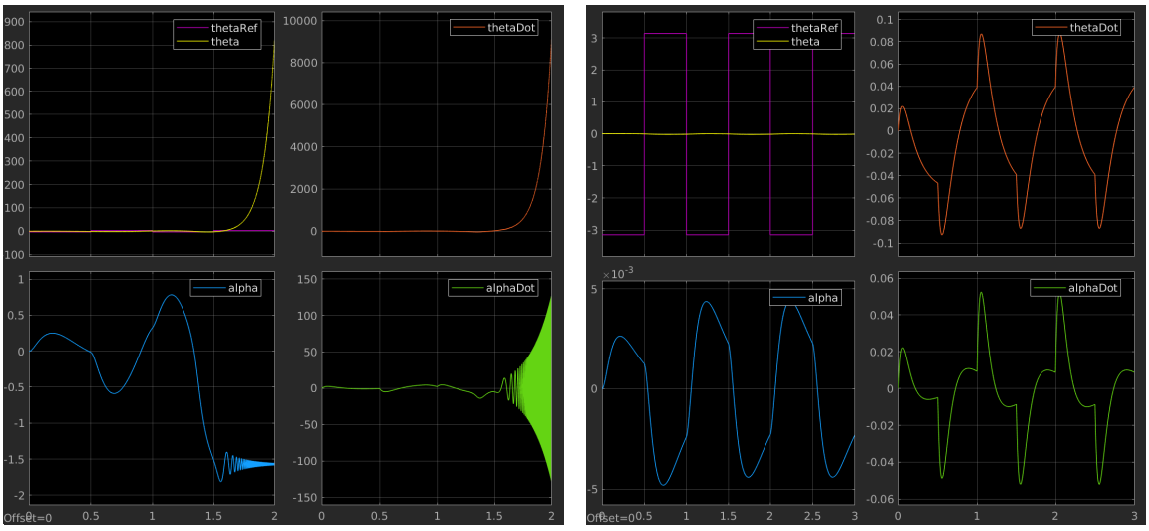

Figure 30: State evolution over time with $K_{u,pp}$ and $K_{u,lqr}$ controller, $\theta_{ref} = \pm 180°$.

As we can expect, since our controller try to follow the square wave reference for theta, it ends up breaking the stability, while the LQR one is too slow and does not follow almost completely the signal reference.

Low Frequency (f=0.1 Hz), α changes. As

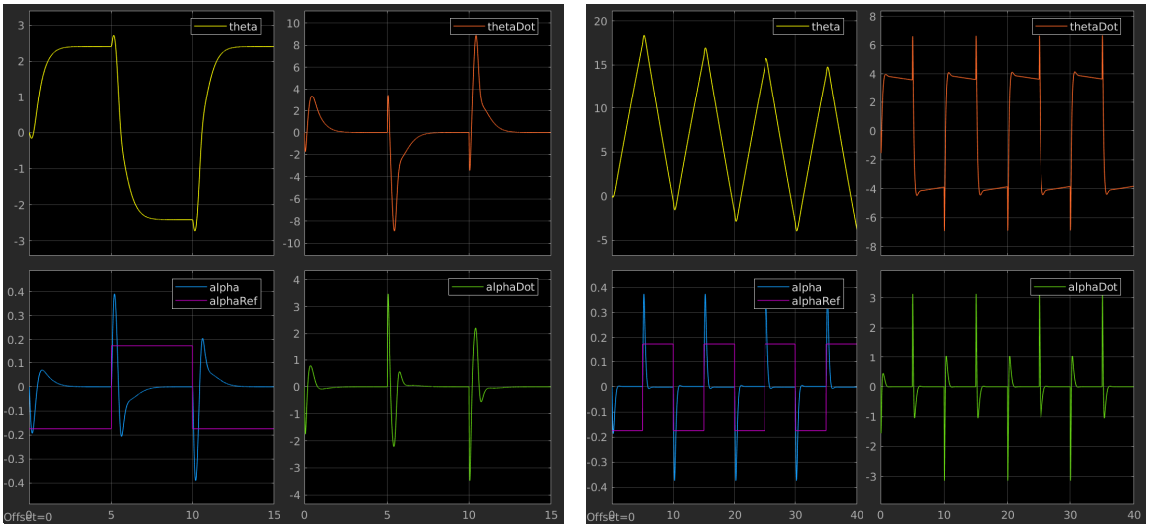

Figure 31: State evolution over time with $K_{u,pp}$ and $K_{u,lqr}$ controller, $\alpha_{ref} = \pm 10°$.

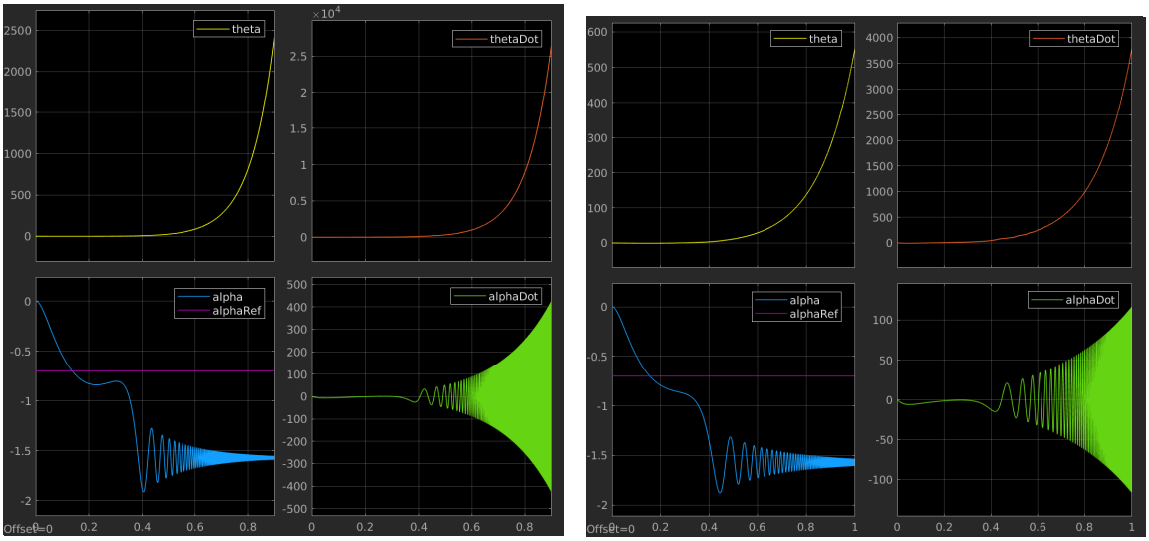

Figure 32: State evolution over time with $K_{u,pp}$ and $K_{u,lqr}$ controller, $\alpha_{ref} = \pm 40°$.

High Frequency (f=1 Hz), α changes. At this frequency, at 10° are stable but we oscillates a lot

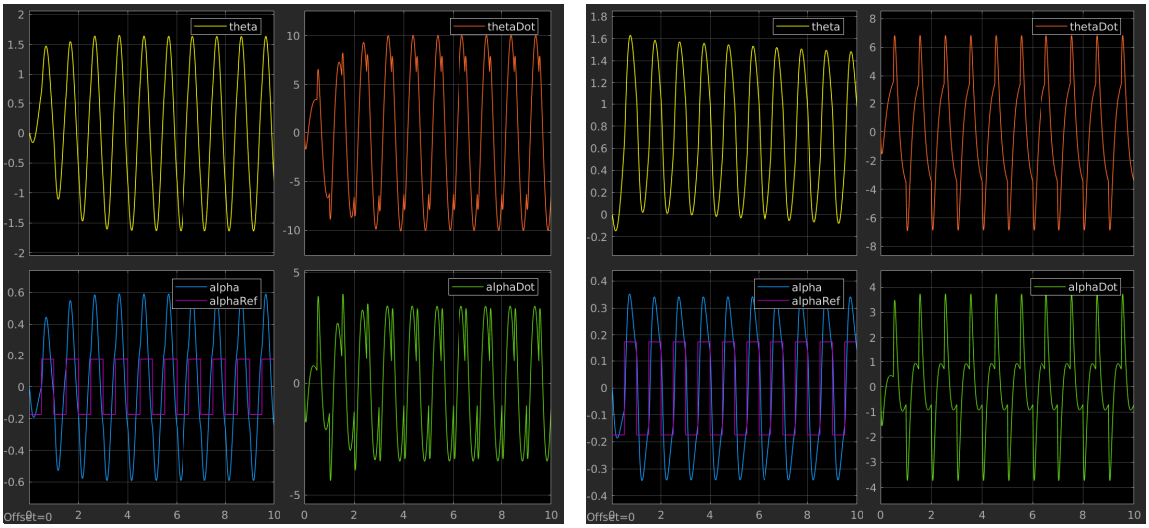

Figure 33: State evolution over time with $K_{u,pp}$ and $K_{u,lqr}$ controller, $\alpha_{ref} = \pm 10°$.

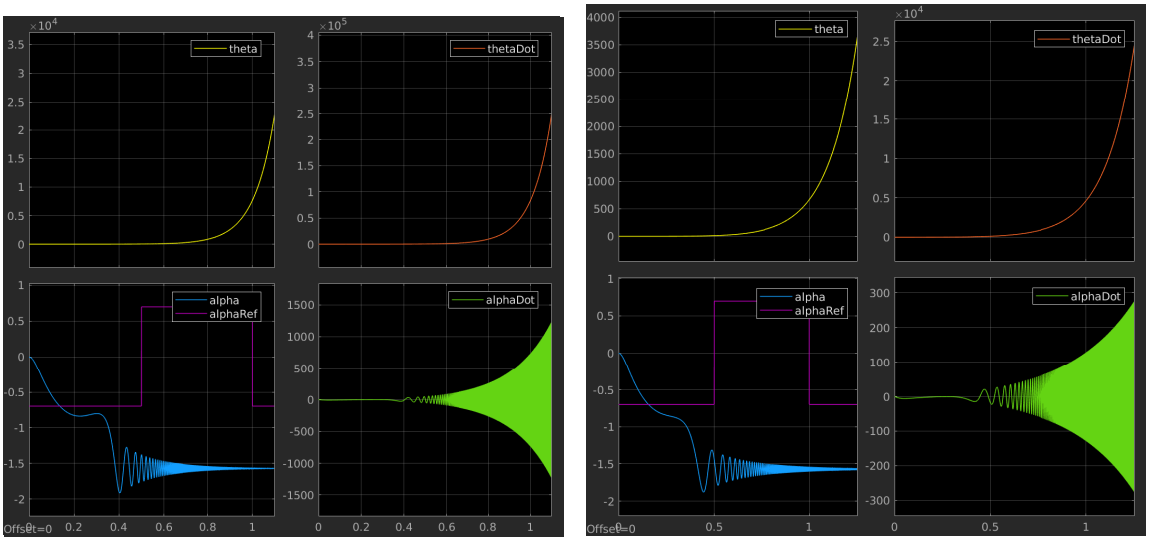

Figure 34: State evolution over time with $K_{u,pp}$ and $K_{u,lqr}$ controller, $\alpha_{ref} = \pm 40°$.

Very High Frequency (f=10 Hz), α changes. As

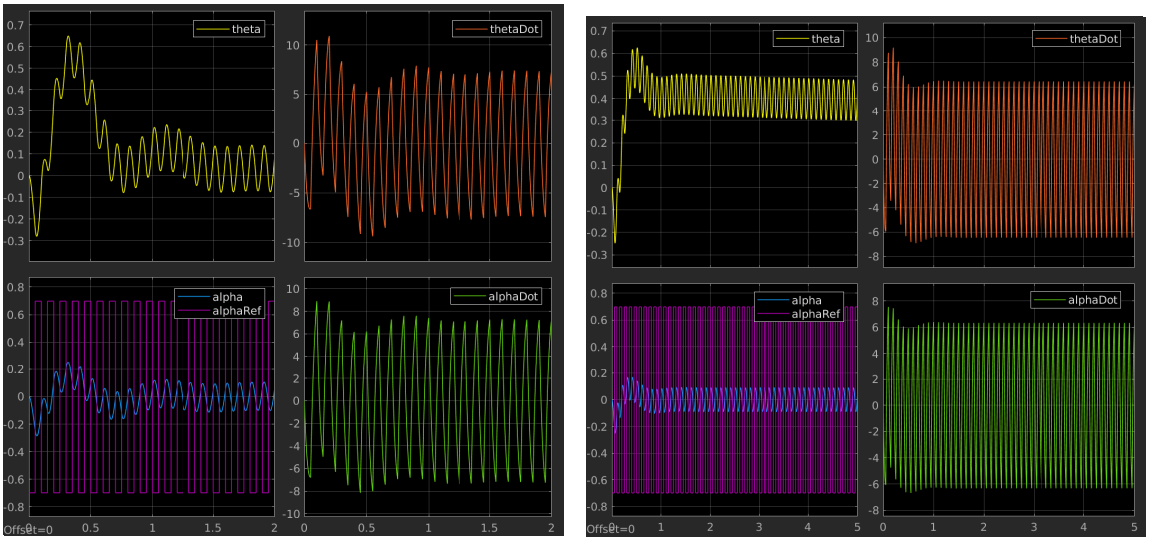

Figure 35: State evolution over time with $K_{u,pp}$ and $K_{u,lqr}$ controller, $\alpha_{ref} = \pm 40°$.

In all the cases we can see how the theta evolution follow the strength imposed by the controller; moreover, we have two cases for which the NL system behaves like the linearized one:

- when the α reference angle is low: in this case, for any frequence, the behaviour is similar, as expected;
- when the frequency is high: in this case, the system, acting like a Low Pass Filter, is not able to reach a sufficiently large state that would bring it to instability; as the frequency increases, the allowed α angle does the same. reaching also previously unstable angles.

The difference from the linearized system is that there we had a controllable couple, so by definition we can reach the origin from any state. in this non linear case, we have instead (??? states that are equivalent to others, so the state is not euclidean and we cannot apply the rules previously found).